WAVELET-BASED VOLTERRA SERIES REPRESENTATIONS OF NONLINEAR DYNAMICAL SYSTEMS

By

RICHARD J. PRAZENICA

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2002

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Andy Kurdila, for his outstanding mentorship. He has been a great advisor, teacher, and friend, and I was very fortunate that he came to the University of Florida five years ago. I would also like to thank Dr. Rick Lind for his advice and many valuable contributions to my research. This includes providing me with the ATW flutter data for the last example in Chapter 7. It has been a great pleasure working with Dr. Kurdila and Dr. Lind. I have learned a great deal from them and hope we continue to work together in the future. I would like to thank Dr. Lou Cattafesta for several productive conversations regarding my research and for serving on my committee. In addition, I would like to thank Drs. Norman Fitz-Coy, Ulrich Kurzweg, and Shari Moskow for taking the time and effort to serve on my committee. Their contributions are much appreciated. I would also like to acknowledge the financial support I received from Dr. Walt Silva and the NASA Langley Research Center.

There are many people within the AeMES Department who have helped make my experience here both enjoyable and memorable. The staff has been tremendous, and I would particularly like to thank Mark Riedy, Shirley Robinson, and Jan Rockey for all their help. I would like to thank Dr. Richard Fearn for his advice and support over the last six years. I learned a great deal working for him as a teaching assistant. I appreciate the many friends I have made here, especially Dwayne McDaniel, Mike Owens, and Andy Sinclair. Their presence made everyday life in the office much more entertaining.

I would like to thank my entire family for their love and support over the years. My mother and grandmother supported the idea of me attending graduate school

from the start and have been there for me every step of the way. I appreciate my brother Kevin for his friendship and have a lot of great memories of the years we spent here together. Finally, I would like to thank my girlfriend Angel for being a constant source of love and encouragement. She has helped me a great deal in seeing this through to the finish and I dedicate this dissertation to her.

TABLE OF CONTENTS

page

ACKNOWLEDGMENTS ii

ABSTRACT vi

1 INTRODUCTION 1

1.1 Volterra Series Representations 5
1.2 Wavelets and Multiresolution Analysis 11

2 FUNDAMENTALS OF VOLTERRA SERIES 19

2.1 General Concepts 20
2.2 The First-Order Volterra Operator 27
2.3 Higher-Order Volterra Operators 33
2.4 Volterra Series Representations of Ordinary Differential Equations 40
2.5 Existence, Uniqueness, and Convergence Issues 50

3 REVIEW OF WAVELETS 55

3.1 Wavelets and Multiresolution Analysis 55
3.2 Wavelets and the Approximation of Functions 65
3.2.1 Fourier Analysis 66
3.2.2 Finite Element Analysis 71
3.2.3 Wavelets 74
3.3 The Haar Wavelet 77
3.4 Orthonormal Wavelets 93
3.5 Orthonormal Multiwavelets 104
3.6 Tensor Product Wavelets 112

4 MULTIWAVELET CONSTRUCTIONS 122

4.1 Multiwavelets and Multiresolution 123
4.2 Construction of Intertwining Multiresolution Analyses 126
4.3 Multiwavelets From Second-Order Finite Elements 130
4.4 Multiwavelets From Third-Order Finite Elements 140
4.5 Piecewise-Polynomial Multiwavelets of Arbitrary Order 153
4.6 Boundary-Adapted Multiwavelets 157
4.7 Multiwavelet Decomposition Example 164

5 WAVELETS SUPPORTED ON TRIANGULAR DOMAINS 169

5.1 Construction of Triangular Wavelets 169
5.2 Triangular Wavelet Decomposition and Reconstruction 185

6 WAVELET-BASED IDENTIFICATION OF VOLTERRA KERNELS . . 193

6.1 Kernel Identification Techniques 193
6.2 Kernel Identification Using Triangular Wavelets 205
6.3 Multiwavelet Kernel Identification 218
6.4 The Least-Squares Problem 227
6.5 Implementation 234

7 NUMERICAL RESULTS 240

7.1 Linear Oscillator 240
7.2 Quadratic Nonlinear System 258
7.3 Nonlinear Oscillator 272
7.4 Flutter Analysis 291

8 CONCLUSION 302

REFERENCES 306

BIOGRAPHICAL SKETCH 316

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

WAVELET-BASED VOLTERRA SERIES REPRESENTATIONS OF NONLINEAR DYNAMICAL SYSTEMS

By

Richard J. Prazenica

August 2002

Chair: Dr. Andrew J. Kurdila
Major Department: Aerospace Engineering, Mechanics and Engineering Science

In this dissertation, reduced-order models for a large class of nonlinear dynamical systems are derived in terms of Volterra series. In a Volterra series representation, the output of a dynamical system is expressed in terms of an infinite series of integral operators. Each Volterra operator has an associated kernel, and these kernels completely characterize the system dynamics. In this study, wavelets, which have proven effective in compressing certain types of integral operators, are used to obtain low-order estimates of first and second-order Volterra kernels for time-invariant systems. Two wavelet bases are constructed for the representation of Volterra kernels. First, piecewise-polynomial multiwavelets are constructed from classical finite element basis functions using the technique of intertwining. These multiwavelets are used to approximate the first-order, linear kernel and the symmetric form of the second-order kernel. The second wavelet basis consists of triangular wavelets that are constructed over a triangular domain. These wavelets are suitable for the representation of the triangular form of the second-order kernel. In this implementation, the Haar wavelet is used to approximate the first-order kernel.

The wavelet-based kernel identification algorithms are validated for both a linear oscillator and a quadratic nonlinear system. The dynamics of these two systems can be completely described in terms of the first and second-order kernels, respectively. For these simple examples, it is demonstrated that relatively accurate, low-order estimates of the analytical kernels can be obtained. First and second-order kernels are then identified for a simulated nonlinear oscillator with a polynomial nonlinearity. It is demonstrated that the identified second-order kernels are capable of predicting the nonlinear dynamics for a bounded range of input amplitudes. For inputs exceeding this amplitude bound, higher-order kernels are needed to model the nonlinear response. Finally, first-order kernels are extracted from flight flutter data from the Aerostructures Test Wing. This example demonstrates the ability of the wavelet-based algorithms to extract first-order kernels from noisy flight data, and a practical application of these kernels to flutter analysis is described.

CHAPTER 1
INTRODUCTION

The development of models for nonlinear dynamical systems is of critical importance in many engineering fields, such as structural dynamics, fluid mechanics, aeroelasticity, biomedical engineering, signal processing, and control theory. Although linear models are often obtained through the linearization of a nonlinear system about some nominal condition, it is often necessary to generate models that include nonlinear effects. This is the case with many aeroservoelastic systems, for example, in which nonlinearities arising from the interaction of the structure, airflow, and flight control system can have a significant effect. Sometimes, as is often the case in biomedical applications, nonlinear models are desired in order to gain a better physical understanding of the system. In other cases, such as in many electrical engineering applications, the purpose of the models is to compensate for nonlinear effects such as signal distortion. In many cases, nonlinear models are sought for the purpose of control design. Due to the computational costs of developing and implementing nonlinear models, it is desirable to obtain reduced-order models whenever possible. This is especially true for the design of online control systems where low-order models are essential. The main objective of this dissertation is to develop reduced-order models that are suitable for the representation of a large class of nonlinear dynamical systems.

In this dissertation, nonlinear models for dynamical systems are generated in the form of Volterra series representations. This approach utilizes the Volterra theory of nonlinear systems, which states that the output of a nonlinear system can be expressed in terms of an infinite sum of integral operators of increasing order. Each

of these operators has an associated kernel, and a given dynamical system is completely characterized by these kernels. Therefore, the problem of system identification becomes one of extracting these Volterra kernels from input/output data from the system. In practice, this is a formidable task due to the large dimensions of the kernels of second order and higher. In general, for an N-dimensional input/output data set, the identification of the rth-order Volterra kernel requires solving for N^r coefficients. The approach detailed in this dissertation uses wavelets and multiresolution analysis to generate reduced-order representations of Volterra kernels. Wavelets have proven to be very effective in compressing certain integral operators [1]. A wavelet basis is composed of the scaled translates and dilates of a single, compactly-supported wavelet function. Therefore, using wavelets and multiresolution analysis, the Volterra kernels can be decomposed in terms of functions that are localized in the time and frequency domains. Often, many of the coefficients in a wavelet expansion are very small and can be neglected, leading to a relatively sparse representation. There are numerous wavelet families from which to choose, each with its own unique set of properties. One of the difficulties associated with using wavelets to represent Volterra kernels is that many wavelets are not easily adapted to the boundaries over which the kernels are supported. A large portion of this dissertation is dedicated to the fundamentals of wavelets and the design of wavelets that are suitable for kernel identification. In particular, the derivation of piecewise-polynomial multiwavelets and the construction of wavelets that are supported on a triangular domain are discussed in detail. The multiwavelets are easily adapted to the square domain over which the symmetric form of the second-order kernel is supported, while the triangular wavelets are well-adapted to the region of support for the triangular form of the second-order kernel. This dissertation focuses on the identification of first and second-order Volterra kernels for time-invariant, single-input/single-output systems. In many cases, the first and second-order kernels alone are sufficient to characterize the system, so the approach

taken here is applicable to a large class of dynamical systems. The wavelet-based kernel identification algorithms are demonstrated on several dynamical systems for which the kernels have a known, analytical form. In addition, they have been implemented in the identification of Volterra kernels from flight flutter data taken from the Aerostructures Test Wing (ATW) at NASA Dryden.

In the remainder of this chapter, a brief introduction and review of literature is given for Volterra theory and wavelets. In Chapter 2, the fundamentals of Volterra series representations are discussed in detail. This chapter includes a discussion of the general properties of Volterra series, the first-order (linear) and higher-order (nonlinear) Volterra kernels, and the relationship between Volterra series and ordinary differential equations (ODEs). Chapter 3 is a review of wavelets and multiresolution analysis. An introduction to wavelets is given, and wavelets are compared to two of the most popular tools for the approximation of functions in engineering practice, Fourier analysis and the finite element method (FEM). The Haar wavelet is discussed in detail as a prototype wavelet for which many wavelet properties are easy to visualize. The discussion is then extended to orthonormal wavelet families and, even more generally, orthonormal multiwavelets. Finally, two-dimensional tensor product wavelets are introduced as a means for approximating two-dimensional functions or signals. Chapter 4 is devoted to the construction of orthonormal, piecewise-polynomial multiwavelets. The method of intertwining multiresolution analyses, developed by Donovan, Geronimo, and Hardin [2, 3], is used to construct piecewise-quadratic and piecewise-cubic multiwavelets. It is demonstrated that this technique can be used to generate multiwavelets of arbitrary approximation order from the classical finite element basis functions. In Chapter 5, the method of constructing wavelets over two-dimensional, finite sets given by Micchelli and Xu [4, 5] is used to create wavelets that are supported over a triangular domain. This domain corresponds to the region over which the triangular form of the second-order Volterra kernel is supported.

The first five chapters constitute the background needed to apply wavelets to the identification of Volterra kernels for dynamical systems. In this dissertation, two different wavelet bases are used for approximating first and second-order Volterra kernels. The piecewise-polynomial multiwavelets derived in Chapter 4 are used to identify the first-order kernel and the symmetric form of the second-order kernel, which is supported on a square domain. The triangular wavelets constructed in Chapter 5 are used to identify the triangular form of the second-order kernel. The triangular wavelets are piecewise-constant in form and are a two-dimensional analog to the Haar wavelet. Therefore, the Haar wavelet and the triangular wavelet basis are used together to identify the first and second-order kernels, respectively. These wavelet-based kernel identification algorithms are described in detail in Chapter 6. A survey of different approaches to kernel identification is given in Section 6.1. Then, the two wavelet implementations are derived in Sections 6.2 and 6.3. In both cases, a matrix formulation is obtained whereby the wavelet coefficients that represent the kernels must be solved for in a least-squares sense. This problem is frequently ill-posed, however, since kernel identification constitutes an inverse problem in that the system is identified from input/output data. Therefore, regularization techniques such as the truncated singular value decomposition are often needed to obtain stable kernel estimates. This is described in Section 6.4. Section 6.5 then elaborates on some of the specific details and issues involved with the implementation of the wavelet-based kernel identification algorithms. Finally, in Chapter 7, these algorithms are used to extract Volterra kernels from input/output data from a number of dynamical systems. First, a linear oscillator is considered in Section 7.1. Then, a nonlinear system that can be described in terms of the second-order kernel alone is analyzed in order to validate the second-order part of the kernel identification algorithms. In Section 7.3, first and second-order kernels are identified for a nonlinear oscillator. The ability of the estimated kernels to predict the outputs corresponding to a number of different

inputs is then evaluated. Section 7.4 gives a practical application of the wavelet-based implementations as Volterra kernels are extracted from flight flutter data taken from the Aerostructures Test Wing (ATW). The ATW was a small-scale wing that was developed at the NASA Dryden Flight Research Center to study flutter, an aeroelastic instability. It is shown that the first-order kernel identified from the flight data can be incorporated into the flutterometer [6], a flutter prediction tool, in order to obtain more accurate flutter estimates. Finally, conclusions and ideas for future research are discussed in Chapter 8.

1.1 Volterra Series Representations

The Volterra theory of nonlinear systems states that the output of a nonlinear dynamical system can be represented by an infinite series of integral operators of increasing order. Each operator has an associated kernel, and these kernels form the model for a given system. The first-order kernel represents linear dynamics and, for a linear system, is equivalent to the impulse response of the system. The first-order operator is simply a convolution of the system input and the first-order kernel. In formulating his theory, Volterra [7] basically extended the concept of convolution to higher dimensions. The resulting operators are capable of describing nonlinear dynamics of varying order. Just as the first-order kernel can be interpreted as an impulse response, the higher-order kernels can be related to the response of the system to multiple impulses applied at different times. It has been noted by many researchers that the Volterra series can be viewed as a polynomial representation for systems with memory, or systems for which the output is influenced by past inputs. As discussed in detail by Boyd *et al.* [8], Volterra series representations are appropriate for systems with fading memory; that is, systems for which the influence of an input applied at a given time fades in a finite period of time. Ku and Wolf [9] determined many of the fundamental convergence properties of Volterra series, which have also been addressed by Boyd *et al.* [10]. Sandberg [11] added to these results in addressing the convergence

of the "doubly finite" Volterra series, which corresponds to the case where not only the series is truncated but also the kernels are assumed to have truncated forms. Fliess *et al.* [12] also provided several important analytical results, such as the derivation of analytical expressions for the Volterra kernels that appear in the representation of ordinary differential equations (ODEs). Wiener [13] developed an orthogonalization of the Volterra series, and many researchers model dynamical systems in terms of the resulting Wiener kernels. There are many texts available that discuss the Volterra theory in detail, including works by Boyd [14], Rugh [15], and Schetzen [16].

Volterra series representations have been used to model nonlinear systems for a diverse array of engineering applications. For many systems of interest, linear models are not sufficient to characterize the dynamics. Volterra series provide a convenient framework for describing the nonlinear behavior of these systems. In many cases, the purpose of these nonlinear models is to gain a better fundamental understanding of the system and to be able to predict the system response to arbitrary excitations. Volterra series have been used extensively in the field of biomedical engineering, for example, to model various physiological systems. Volterra models have been employed to predict the responses of mechanoreceptors, which convey sensory input signals to the central nervous systems of vertebrates and some invertebrates [17], [18], [19]. Developing models for mechanoreceptors, which exhibit complex nonlinear behavior, is important in order to better understand the workings of skeletal motor control systems and may be useful in the design of machines that can interact with the physical world. Volterra models have also been used to model sensory systems that exhibit nonlinear feedback, such as auditory and visual systems [20]. Such models may be able to explain certain pathologies such as tinnitus, or ringing in the ears. Other applications of Volterra series in the biomedical field include characterizing the nonstationary behavior of the rabbit hippocampus [21], describing the viscoelastic response of the human lung [22], modeling the renal autoregulation of blood flow in

rats [23], [24], [25], and developing constitutive equations for soft tissues under axial tension [26].

Within the field of aerospace engineering, Volterra series have recently been employed in the study of aeroelasticity, the interaction of the wing structure of an aircraft with the surrounding airflow. Silva [27], [28] identified Volterra kernels from computational fluid dynamics (CFD) simulations in order to model the response of an aircraft to unsteady aerodynamics at transonic speeds. In aeroelasticity, much attention is directed towards the prediction of flutter, a linear structural instability that occurs when two modes of vibration interact in such a manner that one of the modes becomes undamped. The result is that any small disturbance from the airflow causes unstable vibration of the wing, which is usually catastrophic. Several researchers have recently applied the Volterra theory to predict the onset of flutter and study aeroelastic responses [29], [30], [31], [32], [33], [34]. Limit cycle oscillation (LCO) is a nonlinear aeroelastic phenomenon that leads to periodic vibrations of the wing. LCO is often not catastrophic, but can result in unnecessary fatigue of the aircraft or render an aircraft incapable of performing its mission. LCO is less understood than flutter, and researchers are just beginning to explore the modeling of LCO with Volterra series [32], [33].

There are many other applications in which Volterra models are used to obtain a basic understanding of physical systems. Volterra series have been used to solve inverse scattering problems, which entail the determination of the structures of unknown objects through the computerized processing of diffracted waves [35], [36]. Inverse scattering problems arise in such diverse applications as crystal structure determination, medical ultrasound tomography, underground imaging problems, and searching for dinosaur bones in the New Mexico desert. Volterra series have also been applied for velocity estimation, which is essential in the design of motion detectors

for computer vision [37]. In their text, Worden and Tomlinson [38] discuss the application of Volterra theory to modeling nonlinearities in structural dynamics, focusing on systems such as nonlinear clamped beams and automotive shock absorbers. As a final example, in the field of ocean engineering, Volterra models have been used in the analysis of wave loading on offshore structures [39].

In the above examples, Volterra series were used to gain an understanding of the behavior of various nonlinear systems. In many applications, the presence of nonlinearities signifies a problem that must be alleviated. This is often the case in structural dynamics, where, as an example, the nonlinear response of a beam can be indicative of a fatigue crack [38]. Aiordachioaie *et al.* [40] used Volterra kernels to classify various forms of nonlinearities, such as dead zone, hysteresis, saturation, and quantization, for the purpose of fault detection and compensation. In electrical engineering applications, signals are often distorted due to nonlinear effects. A great deal of research in electrical engineering has been directed towards the development of Volterra filters that model and compensate for these nonlinearities. Volterra filters have been designed for the cancellation of various forms of noise such as atmospheric, underwater acoustic, and urban vehicle noise [41]. Volterra models have been employed in the optimal design of fiber-optic communications systems, which are frequently subject to signal interference due to fiber nonlinearities [42]. In many wireless communications systems, including microwave systems, nonlinearities in the circuits lead to intermodulation distortion, which corrupts the desired signals. Volterra filters have been used to characterize and attenuate this distortion [43], [44] as well as the distortion that can occur due to temperature-dependent nonlinearities [45]. Similarly, Volterra filters have been developed to reduce the distortion in bandpass filters for radio receivers, where nonlinear effects can render a signal unrecoverable [46]. Distortion in communications systems due to nonlinearities in high-power amplifiers has also been alleviated through the use of Volterra filters [47]. In addition, Volterra series have

been used to achieve nonlinear echo cancellation in telecommunications systems [48]. In the field of biomedical engineering, adaptive Volterra filters have been used to filter stimulus artifacts, which consist of stimulus-dependent, time-varying, nonlinear signals that interfere with the measurement of somatosensory evoked potentials (SEPs) [49]. SEPs comprise a class of signals generated by the central and peripheral nervous systems in response to external stimuli and are important for the diagnosis of neuromuscular disorders.

Frequently, the ultimate objective of Volterra models is to design controllers for nonlinear systems. In contrast to some of the previous examples, where Volterra models were used to filter nonlinear effects, in these cases Volterra models are used to generate input signals in order to control system responses. One such example is active noise control (ANC), in which the objective is to cancel the effect of a primary noise source through destructive interference from a controlled secondary source. Active noise control has been applied to cancel noise in such systems as heating, ventilating, and air conditioning (HVAC) systems, headsets, and airplanes. Volterra series have been used to model nonlinearities in primary noise sources and within the control systems in order to achieve better performance of ANC systems [50], [51]. Doyle, Pearson, and Ogunnaike [52] have devoted an entire text to the development of Volterra models for process control. In the text, as well as in a series of papers [53], [54], [55], the authors have outlined nonlinear model predictive control (NMPC) strategies based on Volterra models and applied them to various chemical processes. NMPC is a natural extension of linear model predictive control (MPC), which entails the optimization of an objective function subject to constraints, to nonlinear systems. The Volterra approach allows the nonlinear terms to be incorporated as correction loops in the NMPC system. The controller consists of these nonlinear correction loops in addition to an optimal linear controller. As another example, Volterra series have been used to model the equations of motion of a pendulum in

order to achieve tracking control via feedback linearization [56]. Volterra models have also been employed in the design of exact model matching control systems that are capable of matching a reference model [57]. There are countless other examples. At this point, it should be clear that Volterra series representations are applicable to a large number of diverse systems from virtually every engineering field.

In applying Volterra series representations to generate models for dynamical systems, the problem reduces to one of identifying the Volterra kernels of a given system. Once the kernels are known, the system response to any arbitrary input can be computed from the Volterra operators. Kernel identification is not an easy task due to the large dimensions of the higher-order kernels and the fact that the identification problem is ill-posed. Many different approaches to kernel identification have been taken in the literature. It should be noted that, in theory, any ODE can be converted into an integral equation and solved for its Volterra kernels [12], [15]. This is demonstrated in Chapter 2. Unfortunately, this approach is usually not practical due to high computational costs. In addition, it requires that a given system be modeled by an ODE, which is a serious limitation. In theory, Volterra kernels can be measured by applying impulses to a continuous-time system. Silva used this approach for discrete-time systems by applying unit sample inputs, the discrete analog to impulse inputs, to certain computational fluid dynamics (CFD) models [27, 28]. This technique is not directly applicable to experimental systems, however. Methods for identifying Volterra kernels from experimental data include a white noise/cross correlation method developed by Lee and Schetzen [58] and a harmonic probing technique employed by Boyd *et al.* [59]. Both these approaches are limited, however, in that they require specific input excitations which are not always physically realizable. Another common approach to kernel identification is to approximate the kernels in terms of a set of basis functions. Marmarelis [60] estimated Volterra kernels for biological systems in terms of Laguerre polynomials

and Reisenthel [34] used decaying exponentials to approximate the first and second-order kernels of nonlinear aeroelastic systems. These approaches allow for physically realizable inputs and are, therefore, suitable for experimental systems. Nikolaou and Mantha [61] applied biorthogonal wavelets to compress the first and second-order kernels that model a particular chemical reaction. Kurdila *et al.* [62] also used biorthogonal wavelets for the approximation of Volterra kernels for an experimental aeroelastic system. Neural networks have also been used to estimate Volterra kernels, and Wray and Green [63] have demonstrated that many artificial neural networks have an equivalent infinite Volterra series representation. The kernel identification problem is discussed in more detail in Chapter 6.

1.2 Wavelets and Multiresolution Analysis

The field of wavelets and multiresolution analysis has had a profound impact on a number of disciplines, including mathematics, physics, geosciences, meteorology, medical imagery, and engineering. Although they initially served predominantly as a signal processing tool, wavelets are currently employed in such diverse applications as operator theory, approximation theory, reduced-order modeling of dynamical systems, particle image velocimetry (PIV), and the solution of partial differential equations (PDEs). This diversity can be attributed to the fact that there is a great deal of flexibility in wavelet design. Wavelet bases can be developed that have varying properties such as orthogonality, smoothness, approximation order, and compact support. As wavelets continue to evolve, the list of potential applications will continue to grow.

The term "wavelet" was coined by Meyer, Morlet, and Grossman in 1982. As the name implies, wavelets resemble small waves in that they have the oscillatory character of waves and most practical examples are compactly supported. For this reason, these functions were called "ondelettes," which translates into English as "wavelets." Generally speaking, a wavelet can be defined as any function $\psi \in L^2(\mathbb{R})$ whose family of dilates and translates $\psi_{j,k}(x) \equiv 2^{j/2}\psi(2^j x - k)$ forms a basis for

$L^2(\mathbb{R})$, the Hilbert space of square-integrable functions. Hence, a wavelet expansion of a function or signal affords localization in both the time and frequency domains. Wavelets are constructed from the scaled translates and dilates of a single scaling function, or generator. As will be seen later, this generalizes to multiwavelets, which are constructed from multiple generators. Wavelets are unique from many other functions in that they are not obtained as the solutions of differential equations, but rather as solutions of the two-scale relationship.

The field of wavelets and multiresolution analysis began to take root in the early 1980's as the synthesis of ideas from several disparate disciplines, including physics, mathematics, engineering, and signal processing. Much of the earliest work in the field was motivated by some of the shortcomings of classical Fourier analysis in which a function or signal is decomposed in terms of sinusoids of varying frequency. In general, Fourier methods are well suited for the analysis of smooth signals whose frequency content does not change with time. However, they are less than ideal for the analysis of signals that exhibit sharp features or have frequencies that evolve with time. This is because Fourier methods employ global basis functions and, therefore, cannot easily yield information in the time domain. This limitation led to the emergence of time-frequency analysis from which wavelets are descended. Some of the earliest efforts in this area were put forth by Wigner [64] in 1932 in his research in quantum mechanics and Gabor [65] in 1946 in his work towards the identification of coherent features in sound. In a similar manner, advancements in the field of signal processing formed part of the foundation for wavelets and multiresolution analysis. In the 1970's, the use of banks of parallel filters, usually in the form of pairs of high and low pass filters with complementary properties, became an important topic. During this time, Croiser [66] and Crochiere [67] made significant contributions toward the development of subband coding and its realization using quadrature mirror filter banks. This field has made a precise study of how a signal can be analyzed by decomposing it into constituent

frequency bands. Smith and Barnwell [68] developed conditions whereby the original signal can be perfectly reconstructed using certain types of quadrature mirror filter banks. It will be shown later that, for orthonormal wavelets, the wavelet transform can be viewed as the implementation of a filter bank with perfect reconstruction qualities.

From the preceding discussion, it should be clear that the foundation for wavelets and multiresolution analysis was laid far before the field gained recognition as a distinct discipline in the 1980's. Indeed, it is now widely recognized that the first wavelet, the Haar function, was constructed in 1910. This function has perhaps the simplest form of any wavelet in existence. Although its utility is limited in most applications because it is a discontinuous function with low approximation order, the Haar function serves as an excellent prototype wavelet because many of the properties inherent to wavelets are easy to visualize for this particular function. In addition, the Haar wavelet has several attractive features in that it can be expressed in closed form, yields an orthonormal basis, is compactly supported, and is efficient to implement. While the Haar function possesses simplicity of form in the time (or spatial) domain, the sinc, or Littlewood-Paley, wavelet takes the form of a characteristic function of an interval in the frequency domain. Therefore, it acts as an ideal bandpass filter and the resulting wavelet decomposition has a particularly simple interpretation. Unfortunately, the sinc scaling function and wavelet are not compactly supported, which is a significant limitation in applications. Hence, the sinc wavelet is of interest only from an academic viewpoint.

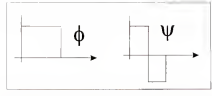
In contrast to the Haar and sinc wavelets, most developments in the field of wavelets and multiresolution analysis center around the derivation of families of wavelets in which the members have specific properties that vary within the family. These properties may include smoothness, approximation order, and length of support. For example, in 1986, Meyer [69] generated a family of wavelets whose

members are in C^k, the set of continuous functions with k continuous derivatives, for k arbitrary but finite. These wavelets lack some desirable characteristics, however, such as compact support, orthonormality, and closed form representations. In 1988, Daubechies [70] constructed a family of wavelets that are compactly supported, orthonormal, and whose functions vary in C^k for k arbitrary but finite. In other words, Daubechies wavelets combine most of the desirable characteristics of the Haar function and Meyer wavelets. This was a revolutionary development that led to the popularization of wavelets for use in the areas of signal processing, statistics, and numerical analysis. However, Daubechies wavelets do not have closed form representations and do not exhibit good symmetry or antisymmetry properties. In 1991, Beylkin, Coifman, and Rokhlin [1] derived a family of wavelets that exhibit similar characteristics in that they are orthogonal, compactly supported, and possess good smoothness and approximation properties. These wavelets have better symmetry properties than Daubechies wavelets. The symmetry properties were achieved via a tradeoff that resulted in wavelets of somewhat larger support. In 1992, Cohen, Daubechies, and Feauveau [71] generated a family of biorthogonal wavelets that has become the most frequently used wavelet family for applications to denoising and compression. By trading orthogonality for biorthogonality, they were able to construct wavelets with better symmetry and antisymmetry characteristics than those derived by Beylkin, Coifman, and Rokhlin while retaining compact support and good smoothness and approximation properties.

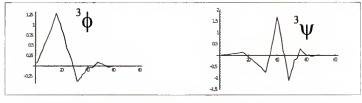
The design of wavelet families always involves tradeoffs in which one particular desired property is obtained at the expense of another. In the mid 1990's, the introduction of multiwavelets, wavelets that are generated from multiple scaling functions, increased the level of available flexibility in wavelet design. Because multiwavelet constructions employ a number of scaling functions and wavelets, the resulting functions

can exhibit a wider array of properties. This does not come without a cost, however, since multiwavelets are typically more costly to implement than their single-generator counterparts. In 1994, Donovan, Geronimo, Hardin, and Massopust [72,73] constructed a class of multiwavelets that are compactly supported, orthonormal, and have good smoothness and approximation properties. These wavelets do not have closed form expressions but rather are fractal in nature. Later, in 1996, they generated a family of piecewise-polynomial multiwavelets having the same properties as well as closed form representations [2,3]. These multiwavelets were derived from the classical finite element basis functions via the technique of intertwining multiresolution analyses. By using the finite element functions of appropriate order, multiwavelets of arbitrary approximation order can be constructed. Donovan, Geronimo, and Hardin [2] carried out this derivation to generate piecewise-linear multiwavelets. In this dissertation, their method is employed to construct piecewise-quadratic and piecewise-cubic multiwavelets. Some of the wavelet bases mentioned in the preceding discussion are shown in Figure (1.1).

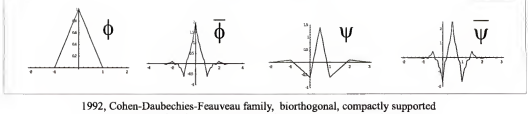
Wavelets have had a profound impact on a number of different disciplines, including mathematics, physics, geoscience, meteorology, medical imagery, and engineering. There are a large number of wavelet texts that are mathematical in nature, such as the works by Daubechies [74], Ogden [75], Chui [76], [77], and Wojtaszczyk [78]. Meanwhile, a great deal of attention has been directed towards wavelet applications. One of the earliest applications of wavelets was by Morlet *et al.* [79] in the study of seismic phenomena encountered in the course of oil prospecting expeditions. As of the early 1980's, wavelets starting becoming popular in signal processing applications such as denoising, data compression, and two-dimensional image compression. Wavelets were seen as an attractive alternative to the traditional Fourier series for the analysis of signals that exhibit sharp features or transient behavior. As will be discussed later, this was due in large part to the fact that wavelets afford the localization of signals in



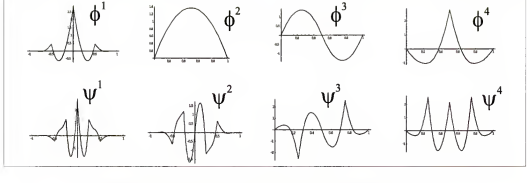
1910, Haar Wavelet, orthonormal, compactly supported
low approximation order, poor smoothness



1988, Daubechies family of wavelets, orthonormal, compactly supported
good approximation order and smoothness, no symmetry or anti-symmetry



1992, Cohen-Daubechies-Feauveau family, biorthogonal, compactly supported
good approximation order and smoothness, symmetry or anti-symmetry



1999, Hardin-Kurdila-Prazenica, orthonormal, compactly supported
good approximation order and smoothness, symmetry or anti-symmetry

Figure 1.1: The Evolution of Wavelet Bases, 1910-2000

both the time and frequency domains. The development of fast algorithms for computing the discrete wavelet transform by Mallat [80] and Meyer [69] was instrumental in the emergence of wavelets as a practical signal processing tool. There are many

texts devoted to the use of wavelets in signal processing, including books by Mallat [81], Strang and Nguyen [82], Vetterli and Kovacevic [83], and Burros, Gopinath, and Guo [84].

As the field of wavelets and multiresolution analysis continued to grow in the late 1980's, the use of wavelets expanded to a diverse array of applications. Wavelets have been implemented in the solution of partial differential equations (PDEs) over smooth manifolds [85]. Similarly, they have been used to solve elliptic boundary-value problems [86] and wavelet-based multilevel methods have been derived for the solution of elliptic, parabolic, and hyperbolic PDEs [87]. Donoho [88] has demonstrated that wavelet systems are nearly optimal for a wide class of parametric estimation problems. In the area of fluid dynamics, wavelet bases have been employed in the analytical study of the Navier-Stokes equations and in a qualitative study of turbulence [89]. Wavelets have also been applied in the post-processing of flow fields from particle image velocimetry (PIV) [90], and divergence-free wavelet bases have been used to obtain reduced-order representations for flow control simulations [91]. In the field of biomedical engineering, wavelets have been employed to remove artifacts and noise from measured electroencephalogram (EEG) signals [92]. Wavelets have also been used for fault detection in mechanical systems [93] and in the characterization of nonlinearities in aeroelastic systems [94]. Also in the field of aeroelasticity, wavelets have been used in the estimation of damping parameters for flutter analysis [95]. As a final example, of most relevance to this dissertation, Beylkin, Coifman, and Rokhlin [1] applied wavelets for the compression of various integral operators. Nikolaou and Mantha [61] used biorthogonal wavelets to compress the first and second-order Volterra kernels corresponding to various chemical processes. Kurdila *et al.* [62] derived reduced-order representations of Volterra kernels in terms of biorthogonal wavelets. This work, as well as the research detailed in this dissertation,

was motivated in large part by the effectiveness of wavelets in compressing integral operators [1], as well as the work of Nikolaou and Mantha [61].

CHAPTER 2
FUNDAMENTALS OF VOLTERRA SERIES

Volterra series representations comprise one of the many tools available for the analysis of nonlinear systems. Volterra series can be applied to either of the two major aspects of nonlinear system theory, analysis and synthesis. In analysis, the goal is to determine the system operator that maps the input into the output while, in synthesis, a system with a given operator is constructed from a set of elemental systems. The focus of this dissertation is on analysis, which entails the identification of the kernels that appear in Volterra series representations of dynamical systems. As will be seen, a given dynamical system is completely characterized in terms of its Volterra kernels. In this chapter, some fundamental properties of Volterra series are reviewed. In Section 2.1, various forms for the Volterra operators under the conditions of time-invariance, causality, and one-sided inputs are examined. Then, in Section 2.2, the first-order Volterra operator is discussed in detail. The first-order operator is sufficient to characterize the dynamics of linear systems and is directly related to classical methods for the analysis of linear, time-invariant systems such as convolution and, in the frequency domain, transfer functions. In Section 2.3, the higher-order Volterra operators that are needed to describe nonlinear system dynamics are discussed. Because the scope of this dissertation is limited to the identification of first and second-order Volterra kernels for nonlinear systems, the focus is on the second order-operator. Various forms for the higher-order kernels, namely the symmetric and triangular forms, are also discussed. Section 2.4 describes the relationship between ordinary differential equations (ODEs) and Volterra series representations. An ODE can be converted into a Volterra integral equation through the process of

integration. Using the method of successive substitutions, also known as Picard iteration, a solution to the resulting integral equation can be derived in terms of a Volterra series. This procedure is demonstrated for both linear and bilinear state space systems, and generalizes to all ODEs. Finally, in Section 2.5, some of the convergence, existence, and uniqueness issues pertaining to Volterra series representations are discussed. Much of the information presented in this chapter has been taken from the texts by Rugh [15] and Schetzen [16]. The interested reader can refer to these texts for a more exhaustive discussion of Volterra theory.

2.1 General Concepts

A system can be viewed as an operator H that maps an input u into an output y:

$$y(t) = H[u(t), t] \tag{2.1}$$

If the system is time-invariant, the operator is no longer an explicit function of time and can be written as $H[u(t)]$. In this discussion, time-invariance is assumed unless otherwise stated. The output of a nonlinear dynamical system can be expressed in terms of a Volterra series, which is an infinite series of integral operators

$$y(t) = \sum_{n=1}^{\infty} y_n(t) = \sum_{n=1}^{\infty} H_n[u(t)] \tag{2.2}$$

In equation (2.2), $y_n(t)$ is the output of the nth-order Volterra operator H_n that, for the time-invariant case, takes the form

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\sigma_1, \ldots, \sigma_n)\, u(t-\sigma_1) \cdots u(t-\sigma_n)\, d\sigma_1 \cdots d\sigma_n \tag{2.3}$$

where h_n is termed the nth-order Volterra kernel of the system. The form of the nth-order Volterra operator shown in equation (2.3) is sometimes referred to as a

generalized convolution integral. It should be noted that a zero-order term y_0 can be added in equation (2.2) to represent systems that have nonzero responses to an identically zero input. For example, the zero-order operator characterizes part of the transient response of a system to nonzero initial conditions. In this dissertation, most of the examples considered have initial conditions of zero so that the zero-order operator vanishes. In addition to being a series of operators which transforms the input function into the output function, a Volterra series representation can also be viewed as a functional series. If one is interested in the output at a particular instant in time t_1, the Volterra series acts as a functional, mapping the input function u into the output at time t_1, $y(t_1)$.

Clearly, a system is characterized by its Volterra kernels since, once the kernels of a particular system are known, the response to any input can be predicted using equations (2.2) and (2.3). It should be noted that the kernels that appear in equation (2.3) are not necessarily unique for a given dynamical system. Indeed, except for the first-order kernel, the nth-order Volterra kernel of a nonlinear system can be expressed in more than one form. However, as will be shown in Section 2.3, a symmetric form for the nth-order Volterra kernel can always be derived that is unique for a given system. The output of certain systems can be expressed in terms of a finite number of Volterra operators in equation (2.2). In some cases, a system is completely characterized by only one Volterra kernel. For example, a linear system can be represented in terms of the first-order Volterra kernel alone, since, as will be discussed later, the first-order Volterra operator is a linear operator. The higher-order Volterra operators, which are nonlinear, do not contribute to the response of a linear system. A system that can be characterized in terms of the nth-order Volterra kernel alone is known as a degree n homogeneous system [15]. The output of such a system is given by

$$y(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\sigma_1, \ldots, \sigma_n)\, u(t-\sigma_1) \cdots u(t-\sigma_n)\, d\sigma_1 \cdots d\sigma_n \qquad (2.4)$$

The term degree n homogeneous system refers to the fact that, given an input of the form αu, where α is a scalar, the output of the system is $\alpha^n y$, where y is the response of the system to the input u. This property is readily apparent from equation (2.4). Some systems can be described in terms of a finite sum of Volterra operators

$$y(t) = \sum_{n=1}^{N} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\sigma_1, \ldots, \sigma_n)\, u(t-\sigma_1) \cdots u(t-\sigma_n)\, d\sigma_1 \cdots d\sigma_n \qquad (2.5)$$

Such systems are sometimes referred to as polynomial systems of degree N, where h_N is the highest-order nonzero kernel needed to characterize the system dynamics.

The output of the nth-order Volterra operator, given by equation (2.3), can be expressed in an alternate form by performing a change of variables:

$$\begin{aligned} \tau_1 &= t - \sigma_1 & d\tau_1 &= -d\sigma_1 \\ &\vdots & &\vdots \\ \tau_n &= t - \sigma_n & d\tau_n &= -d\sigma_n \end{aligned}$$

The limits of integration for each integral become $\tau_1 : \infty \to -\infty, \ \ldots\ , \tau_n : \infty \to -\infty$. Substituting into equation (2.5), we obtain

$$y_n(t) = \int_{\infty}^{-\infty} \cdots \int_{\infty}^{-\infty} h_n(t-\tau_1, \ldots, t-\tau_n)\, u(\tau_1) \cdots u(\tau_n)\, (-d\tau_1) \cdots (-d\tau_n)$$

Noting that the negative sign in each integral changes the order of the integration limits, we have

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(t-\tau_1, \ldots, t-\tau_n)\, u(\tau_1) \cdots u(\tau_n)\, d\tau_1 \cdots d\tau_n \qquad (2.6)$$

Changing variables once again in equation (2.6), it is clear that we have the following equivalent expressions for the output of the nth-order Volterra operator for a time-invariant system:

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\sigma_1, \ldots, \sigma_n)\, u(t-\sigma_1) \cdots u(t-\sigma_n)\, d\sigma_1 \cdots d\sigma_n \tag{2.7}$$

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(t-\sigma_1, \ldots, t-\sigma_n)\, u(\sigma_1) \cdots u(\sigma_n)\, d\sigma_1 \cdots d\sigma_n \tag{2.8}$$

It is not difficult to demonstrate that the Volterra operators in equations (2.7) and (2.8) are time-invariant, or stationary, operators. A time-invariant operator does not vary with time so that a time shift of the input results in an equivalent time shift of the output. Therefore, an operator H defined as

$$y(t) = H[u(t)]$$

is time-invariant, or stationary, if

$$y(t-\tau) = H[u(t-\tau)] \tag{2.9}$$

Checking the expression for the nth-order Volterra operator in equation (2.7) for time-invariance, the time-shifted output $y_n(t-\tau)$ is given by

$$y_n(t-\tau) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\sigma_1, \ldots, \sigma_n)\, u(t-\tau-\sigma_1) \cdots u(t-\tau-\sigma_n)\, d\sigma_1 \cdots d\sigma_n \tag{2.10}$$

The operator H_n acting on the time-shifted input $u(t-\tau)$ yields

$$H_n[u(t-\tau)] = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\sigma_1, \ldots, \sigma_n)\, u(t-\tau-\sigma_1) \cdots u(t-\tau-\sigma_n)\, d\sigma_1 \cdots d\sigma_n \tag{2.11}$$

Clearly, then, comparing equations (2.10) and (2.11),

$$y_n(t-\tau) = H_n[u(t-\tau)] \tag{2.12}$$

and it follows that H_n is a time-invariant operator. Nonstationary systems can also be represented using Volterra series. In this case, the expression for the nth-order Volterra operator in equation (2.8) takes the form

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(t, \sigma_1, \ldots, \sigma_n)\, u(\sigma_1) \cdots u(\sigma_n)\, d\sigma_1 \cdots d\sigma_n \tag{2.13}$$

where $h_n(t, \sigma_1, \ldots, \sigma_n)$ is a nonstationary kernel. It follows that for a given kernel $h_n(t, \sigma_1, \ldots, \sigma_n)$ to be stationary, there must exist a kernel $g_n(t_1, \ldots, t_n)$ such that

$$h_n(t, \sigma_1, \cdots, \sigma_n) = g_n(t-\sigma_1, \cdots, t-\sigma_n) \tag{2.14}$$

where $t_1 = t - \sigma_1, \ \ldots \ , t_n = t - \sigma_n$. Rugh [15] has demonstrated that a kernel $h_n(t, \sigma_1, \ldots, \sigma_n)$ can be checked for stationarity by time-shifting its arguments by t. If the relationship

$$h_n(t, \sigma_1, \cdots, \sigma_n) = h_n(0, \sigma_1 - t, \cdots, \sigma_n - t) \tag{2.15}$$

is satisfied, then the kernel is stationary and $g_n(t_1, \ldots, t_n)$ is obtained as

$$g_n(t_1, \cdots, t_n) = h_n(0, -t_1, \cdots, -t_n) \tag{2.16}$$

Hence, equation (2.15) is a convenient means for testing a given kernel for time-invariance [15]. If this condition is satisfied, equation (2.16) gives the stationary form of the kernel that would appear in equation (2.8).

Frequently, dynamical systems are assumed to be causal, which means that the value of the present output is not dependent on future inputs. Intuitively, it is apparent that all physical systems are causal. Recall that the output of the nth-order Volterra operator for a time-invariant dynamical system can be written in the form

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\sigma_1, \ldots, \sigma_n)\, u(t-\sigma_1) \cdots u(t-\sigma_n)\, d\sigma_1 \cdots d\sigma_n \qquad (2.17)$$

It is clear from equation (2.17) that, given the present time t, if $\sigma_i < 0$ for any value of $i = 1, \ldots, n$, then the corresponding input $u(t-\sigma_i)$ represents a future input at time greater than t. It follows that in order for causality to hold, we must have

$$h_n(\sigma_1, \cdots, \sigma_n) = 0 \quad \text{for any } \sigma_i < 0, \quad i = 1, \ldots, n \qquad (2.18)$$

This condition reflects the fact that future inputs have no influence on the present output. Similarly, if the time-invariant Volterra operator is in the alternate form

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(t-\sigma_1, \ldots, t-\sigma_n)\, u(\sigma_1) \cdots u(\sigma_n)\, d\sigma_1 \cdots d\sigma_n \qquad (2.19)$$

it is clear that, given the present time t, if $\sigma_i > t$ for any value of $i = 1, \ldots, n$, then the corresponding input $u(\sigma_i)$ represents an input at time greater than t. Then, it follows that causality requires

$$h_n(t-\sigma_1, \cdots, t-\sigma_n) = 0 \quad \text{for any } \sigma_i > t, \quad i = 1, \ldots, n \qquad (2.20)$$

The causality conditions in equations (2.18) and (2.20) can be directly incorporated into the corresponding expressions for the Volterra operators by changing the limits of integration. Equation (2.18) implies that, for a causal system, the lower limits of integration in equation (2.17) can be changed to zero:

$$y_n(t) = \int_0^{\infty} \cdots \int_0^{\infty} h_n(\sigma_1, \ldots, \sigma_n)\, u(t-\sigma_1) \cdots u(t-\sigma_n)\, d\sigma_1 \cdots d\sigma_n \qquad (2.21)$$

since h_n is zero if any σ_i, $i = 1, \ldots, n$, is negative. Alternatively, the causality condition in equation (2.20) implies that the upper integration limits on each σ_i can be reduced to t in equation (2.19):

$$y_n(t) = \int_{-\infty}^{t} \cdots \int_{-\infty}^{t} h_n(t-\sigma_1, \ldots, t-\sigma_n)\, u(\sigma_1) \cdots u(\sigma_n)\, d\sigma_1 \cdots d\sigma_n \qquad (2.22)$$

It should also be noted that, in practice, one-sided inputs are usually encountered [15]. One-sided inputs are usually characterized by $u(t) = 0$ for $t < 0$. Then, for causal, time-invariant systems with one-sided inputs, the upper limits of integration in equation (2.21) can be reduced to t, while the lower limits of integration in equation (2.22) can be changed to 0. Under these conditions, the nth-order Volterra operator takes either of the following forms:

$$y_n(t) = \int_0^{t} \cdots \int_0^{t} h_n(\sigma_1, \ldots, \sigma_n)\, u(t-\sigma_1) \cdots u(t-\sigma_n)\, d\sigma_1 \cdots d\sigma_n \qquad (2.23)$$

$$y_n(t) = \int_0^{t} \cdots \int_0^{t} h_n(t-\sigma_1, \ldots, t-\sigma_n)\, u(\sigma_1) \cdots u(\sigma_n)\, d\sigma_1 \cdots d\sigma_n \qquad (2.24)$$

Therefore, it is apparent that under certain conditions, the limits of integration in the various expressions for the Volterra operators can take on a number of different

values. In such cases, the integration limits that are most convenient are usually chosen.

As a final note, the results given in this section are applicable to single-input/single-output (SISO) systems. It is not difficult, however, to extend these results to multiple-input/multiple-output (MIMO) systems. For multiple outputs, each output has its own Volterra series expansion with a unique set of kernels. In the case where there are multiple inputs, a given output will have a Volterra series representation consisting of "self-kernels" corresponding to each individual input and "cross-kernels" that involve multiple input functions [96]. As an example, a second-order Volterra series representation of a system with two inputs, u and v, and a single output y is given by

$$\begin{aligned} y(t) &= \int_0^t h_{1,u}(\sigma)u(t-\sigma)d\sigma + \int_0^t \int_0^t h_{2,u,u}(\sigma_1,\sigma_2)u(t-\sigma_1)u(t-\sigma_2)d\sigma_1 d\sigma_2 \\ &+ \int_0^t h_{1,v}(\sigma)v(t-\sigma)d\sigma + \int_0^t \int_0^t h_{2,v,v}(\sigma_1,\sigma_2)v(t-\sigma_1)v(t-\sigma_2)d\sigma_1 d\sigma_2 \\ &+ \int_0^t \int_0^t h_{2,u,v}u(t-\sigma_1)v(t-\sigma_2)d\sigma_1 d\sigma_2 \end{aligned} \tag{2.25}$$

where $h_{2,u,v}$ is the second-order cross-kernel.

2.2 The First-Order Volterra Operator

The output of the first-order Volterra operator for a time-invariant dynamical system can be written in either of the following forms:

$$y_1(t) = H_1[u(t)] = \int_{-\infty}^{\infty} h_1(\sigma)\, u(t-\sigma)\, d\sigma \tag{2.26}$$

$$y_1(t) = H_1[u(t)] = \int_{-\infty}^{\infty} h_1(t-\sigma)\, u(\sigma)\, d\sigma \tag{2.27}$$

The equivalence of these two forms was shown in Section 2.1 for the general nth-order operator. Recall that if the system is causal with one-sided inputs, the integration limits in both expressions can be reduced to 0 to t.

It is straightforward to show that the first-order Volterra operator is a linear operator. A linear operator L is defined as an operator

$$y(t) = L[u(t)]$$

for which a linear combination of inputs results in the same linear combination of the corresponding outputs. In other words, the principle of superposition holds for linear operators. Mathematically, given an input of the form

$$u(t) = \alpha\, v(t) + \beta\, w(t) \tag{2.28}$$

where α and β are scalars, the output y of a linear operator can be written as

$$y(t) = L[\alpha\, v(t) + \beta\, w(t)] = \alpha\, L[v(t)] + \beta\, L[w(t)] \tag{2.29}$$

To demonstrate that the first-order Volterra operator is a linear operator, note that, given an input of the form in equation (2.28), the output of the first-order operator in equation (2.27) is

$$\begin{aligned} y_1(t) = H_1[u(t)] &= \int_{-\infty}^{\infty} h_1(t-\sigma)\{\alpha v(\sigma) + \beta w(\sigma)\}\, d\sigma \\ &= \alpha \int_{-\infty}^{\infty} h_1(t-\sigma)\, v(\sigma)\, d\sigma + \beta \int_{-\infty}^{\infty} h_1(t-\sigma)\, w(\sigma)\, d\sigma \end{aligned}$$

Therefore, we have

$$y_1(t) = H_1[\alpha\, v(t) + \beta\, w(t)] = \alpha\, H_1[v(t)] + \beta\, H_1[w(t)] \tag{2.30}$$

Similarly, given the input in equation (2.28), note that

$$u(t-\sigma) = \alpha\, v(t-\sigma) + \beta\, w(t-\sigma)$$

Then, the output of the first-order Volterra operator in equation (2.26) takes the form

$$\begin{aligned} y_1(t) = H_1[u(t)] &= \int_{-\infty}^{\infty} h_1(\sigma)\{\alpha\, v(t-\sigma) + \beta\, w(t-\sigma)\}\, d\sigma \\ &= \alpha \int_{-\infty}^{\infty} h_1(\sigma)\, v(t-\sigma)\, d\sigma + \beta \int_{-\infty}^{\infty} h_1(\sigma)\, w(t-\sigma)\, d\sigma \end{aligned}$$

and we obtain the same result as in equation (2.30):

$$y_1(t) = H_1[\alpha\, v(t) + \beta\, w(t)] = \alpha\, H_1[v(t)] + \beta\, H_1[w(t)] \tag{2.31}$$

Clearly, then, the first-order Volterra operator is a linear operator. In the next section, it will be shown that the higher-order Volterra operators are nonlinear operators.

The output of a linear system can be completely represented in terms of the first-order Volterra operator, and the system is completely characterized by the corresponding first-order kernel. If the system is also time-invariant, it is classified as a linear time-invariant (LTI) system. It is a well-known classical result that the output of an LTI system can be obtained through the convolution of the input function with the impulse response of the system, where the impulse response is simply the response of the system to a unit impulse applied at $t = 0$. Then, the output of an LTI system is given by the convolution integral

$$y(t) = \int_{-\infty}^{\infty} h_1(\sigma)\, u(t-\sigma)\, d\sigma = \int_{-\infty}^{\infty} h_1(t-\sigma)\, u(\sigma)\, d\sigma \tag{2.32}$$

where h_1 is the impulse response of the system. Clearly, the convolution integral in equation (2.32) is of the same form as the expressions for the output of the first-order Volterra operator in equations (2.26) and (2.27). Therefore, the first-order Volterra kernel h_1 of an LTI system is equivalent to the impulse response of the system.

Often, it is advantageous to analyze a system in the frequency domain. Towards this end, the Fourier transform of the first-order Volterra kernel is defined as

$$H_1(j\omega) = \int_{-\infty}^{\infty} h_1(t)\, e^{-j\omega t} dt \tag{2.33}$$

where, for an LTI system, $H_1(j\omega)$ is termed the frequency response function. The impulse response function can be obtained from the frequency response function using the inverse Fourier transform:

$$h_1(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} H_1(j\omega)\, e^{j\omega t} d\omega \tag{2.34}$$

The convolution operation in the time domain is equivalent to multiplication in the frequency domain. To demonstrate this property, note that the Fourier transform of the output is

$$Y(j\omega) = \int_{-\infty}^{\infty} y(t)\, e^{-j\omega t} dt \tag{2.35}$$

Substituting for the output in terms of the convolution integral, we have

$$Y(j\omega) = \int_{-\infty}^{\infty} \left\{ \int_{-\infty}^{\infty} h_1(\sigma)\, u(t-\sigma)\, d\sigma \right\} e^{-j\omega t} dt$$

Changing variables such that $\tau = t - \sigma$ and rearranging the terms,

$$\begin{aligned} Y(j\omega) &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h_1(\sigma)\, u(\tau)\; e^{-j\omega(\tau+\sigma)} d\sigma\; d\tau \\ &= \int_{-\infty}^{\infty} h_1(\sigma)\, e^{-j\omega\sigma} d\sigma \int_{-\infty}^{\infty} u(\tau)\; e^{-j\omega\tau}\; d\tau \end{aligned}$$

Finally, comparing the terms in the above expression to equation (2.33), we obtain

$$Y(j\omega) = H_1(j\omega)\, U(j\omega) \tag{2.36}$$

Therefore, from equation (2.36), it is clear that convolution in the time domain is equivalent to multiplication in the frequency domain.

The frequency domain representation of a system is especially convenient when the input is periodic. A function u is periodic if there exists a real number T such that $u(t+T) = u(t)$ for all values of t. As noted by Schetzen [16], a periodic input can be expanded in terms of a Fourier series as follows:

$$u(t) = \sum_{n=-\infty}^{\infty} c_n e^{jn\omega_0 t} \tag{2.37}$$

where ω_0 is the fundamental frequency of the input and $\{c_n\}$ are complex Fourier expansion coefficients. For an LTI system, the output takes the form

$$y(t) = \sum_{n=-\infty}^{\infty} c_n \int_{-\infty}^{\infty} h_1(\sigma) e^{jn\omega_0(t-\sigma)} d\sigma \tag{2.38}$$

where the principle of superposition has been applied. Note that, from equation (2.33),

$$H_1(jn\omega_0) = \int_{-\infty}^{\infty} h_1(\sigma)\, e^{-jn\omega_0\sigma} d\sigma \tag{2.39}$$

Then, equation (2.38) can be written as

$$y(t) = \sum_{n=-\infty}^{\infty} c_n H_1(jn\omega_0)\, e^{jn\omega_0 t} = \sum_{n=-\infty}^{\infty} d_n e^{jn\omega_0 t} \tag{2.40}$$

Equation (2.40) shows that, for a periodic input, the output of an LTI system is also periodic and can be represented in terms of a Fourier series where the Fourier expansion coefficients $\{d_n\}$ are given by

$$d_n = H_1(jn\omega_0)\, c_n \tag{2.41}$$

Equation (2.40) also demonstrates that, for an LTI system, the output contains only frequency components that are present in the input. This is not true, in general,

for nonlinear systems. Equation (2.41) shows that the Fourier expansion coefficients of the output are the input coefficients scaled by the frequency response function at each frequency.

It is important to note that the analysis of an LTI system in the frequency domain is contingent upon the existence of a finite frequency response function $H_1(j\omega)$. It can be shown that a sufficient condition for the existence of the frequency response function is that the system be bounded-input, bounded-output (BIBO) stable [16]. An LTI system is BIBO stable provided that, given an input that is bounded such that $|u(t)| < M$ for all t (where M is a finite, real number), the output is also bounded for all t. Note that, for a bounded input,

$$\begin{aligned} |y(t)| = \left| \int_{-\infty}^{\infty} h_1(\sigma)\, u(t-\sigma)\, d\sigma \right| &\leq \int_{-\infty}^{\infty} |h_1(\sigma)|\, |u(t-\sigma)|\, d\sigma \\ &< M \int_{-\infty}^{\infty} |h_1(\sigma)|\, d\sigma \end{aligned} \tag{2.42}$$

In order for the output to be bounded, we must have $|y(t)| < \infty$. Clearly, from equation (2.42), y will be bounded if the following condition is satisfied:

$$\int_{-\infty}^{\infty} |h_1(\sigma)|\, d\sigma < \infty \tag{2.43}$$

Equation (2.43) is a sufficient condition for the boundedness of $y(t)$. To show that equation (2.43) is also a necessary condition, Schetzen [16] noted that a "worst case" input can be defined such that

$$u(t-\sigma) = \begin{cases} +1 & h_1(\sigma) \geq 0 \\ -1 & h_1(\sigma) < 0 \end{cases} \tag{2.44}$$

The input defined in equation (2.44) is bounded since $|u(t-\sigma)| = 1$ for all t. Then, for this input,

$$|y(t)| = \left| \int_{-\infty}^{\infty} h_1(\sigma)\, u(t-\sigma)\, d\sigma \right| = \int_{-\infty}^{\infty} |h_1(\sigma)|\, d\sigma \tag{2.45}$$

In order for y to be bounded, the inequality in equation (2.43) must be satisfied. Therefore, equation (2.43) is both a necessary and sufficient condition for a system to be BIBO stable. The existence of a finite frequency response function for an LTI system can be examined by first noting that, from Euler's identity,

$$\left| e^{-j\omega t} \right| = |\cos(\omega t) - j \sin(\omega t)| = 1 \tag{2.46}$$

Then, using equation (2.33), a finite frequency response function $H_1(j\omega)$ exists if

$$|H_1(j\omega)| = \left| \int_{-\infty}^{\infty} h_1(t)\, e^{-j\omega t} dt \right| \leq \int_{-\infty}^{\infty} |h_1(t)| \left| e^{-j\omega t} \right| dt < \infty \tag{2.47}$$

Therefore, using the result from equation (2.46), a sufficient condition for the existence of a finite frequency response function is

$$\int_{-\infty}^{\infty} |h_1(t)|\, dt < \infty \tag{2.48}$$

Equation (2.48) is exactly the condition that must be satisfied for an LTI system to be BIBO stable. Therefore, an LTI system possesses a finite frequency response function and, equivalently, a finite-valued first-order Volterra kernel, if it is BIBO stable [16].

2.3 Higher-Order Volterra Operators

The representation of nonlinear dynamical systems using Volterra series requires the inclusion of Volterra operators of second order and higher. This dissertation focuses on those nonlinear systems that can be represented in terms of the first and second-order Volterra operators alone. These systems fall under the classification of "weakly nonlinear" systems. This section focus on the properties of the second-order

Volterra operator, noting that the discussion can easily be extended to higher-order operators. This discussion is also limited to time-invariant systems, although it is not difficult to generalize the results to nonstationary systems.

Recall that the output of the nth-order Volterra operator for a time-invariant dynamical system takes either of the following equivalent forms:

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\sigma_1, \ldots, \sigma_n)\, u(t-\sigma_1) \cdots u(t-\sigma_n)\, d\sigma_1 \cdots d\sigma_n \tag{2.49}$$

$$y_n(t) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(t-\sigma_1, \ldots, t-\sigma_n)\, u(\sigma_1) \cdots u(\sigma_n)\, d\sigma_1 \cdots d\sigma_n \tag{2.50}$$

The second-order Volterra operator, then, can be written in either of the forms

$$y_2(t) = H_2[u(t)] = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h_2(\sigma_1, \sigma_2)\, u(t-\sigma_1)\, u(t-\sigma_2)\, d\sigma_1 d\sigma_2 \tag{2.51}$$

$$y_2(t) = H_2[u(t)] = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h_2(t-\sigma_1, t-\sigma_2)\, u(\sigma_1)\, u(\sigma_n)\, d\sigma_1 d\sigma_2 \tag{2.52}$$

where h_2 is the second-order Volterra kernel of the system. It is straightforward to show that the second-order Volterra operator is a nonlinear operator. A nonlinear operator is, by definition, any operator that is not linear. Assuming an input of the form

$$u(t) = \alpha\, v(t) + \beta\, w(t) \tag{2.53}$$

where α and β are scalars, the output of the second order operator in equation (2.52) is given by

$$H_2[u(t)] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_2(t-\sigma_1, t-\sigma_2)\{\alpha v(\sigma_1) + \beta w(\sigma_1)\} \{\alpha v(\sigma_2) + \beta w(\sigma_2)\} d\sigma_1 d\sigma_2 \quad (2.54)$$

Recall that, in order for H_2 to be a linear operator, the following condition must be satisfied:

$$H_2[\alpha v(t) + \beta w(t)] = \alpha H_2[v(t)] + \beta H_2[w(t)] \quad (2.55)$$

Note that

$$\begin{aligned} \alpha H_2[v(t)] + \beta H_2[w(t)] &= \alpha \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_2(t-\sigma_1, t-\sigma_2) v(\sigma_1) v(\sigma_2) d\sigma_1 d\sigma_2 \\ &\quad + \beta \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_2(t-\sigma_1, t-\sigma_2) w(\sigma_1) w(\sigma_2) d\sigma_1 d\sigma_2 \end{aligned} \quad (2.56)$$

It is clear that the expressions in equations (2.54) and (2.56) are not equivalent and equation (2.55) is not satisfied. Therefore, the second-order Volterra operator is a nonlinear operator. The same result could have been obtained using the expression for the second-order operator in equation (2.51). Similarly, the Volterra operators of third order and higher can also be shown to be nonlinear.

Nonlinear dynamical systems can be represented in terms of Volterra series expansions that include Volterra operators of second order and higher. A given system is characterized by the Volterra kernels that appear in this representation. Just as the first-order kernel can be interpreted as an impulse response function, higher-order kernels can be viewed in terms of the system response to multiple impulse inputs. For example, for a degree two homogeneous system, different "components" of the second-order kernel can be related to the system response to two impulses applied at

different times. Indeed, in theory, higher-order Volterra kernels can be measured by applying multiple impulses to the system. This approach is rarely practical, however, mainly due to the difficulty in applying impulse inputs to physical systems.

It should be noted that the Volterra kernels, except for the first-order kernel, are not unique for a given system. It is often convenient, for computational purposes, to impose uniqueness on the Volterra kernels. It can be shown that unique forms for the kernels exist in terms of special restricted forms such as the symmetric, triangular, and regular forms [15]. Here, only the symmetric and triangular forms are discussed. A second-order Volterra kernel h_2 is symmetric if it satisfies

$$h_2(\sigma_1, \sigma_2) = h_2(\sigma_2, \sigma_1) \tag{2.57}$$

It can be shown that any second-order kernel can be converted into the symmetric form shown in equation (2.57). As an example, consider a separable second-order kernel of the form

$$h_2(\sigma_1, \sigma_2) = f(\sigma_1)\, g(\sigma_2) \tag{2.58}$$

This kernel is not necessarily symmetric since, in general,

$$f(\sigma_1)\, g(\sigma_2) \neq g(\sigma_1)\, f(\sigma_2)$$

The kernel in equation (2.58) can be viewed as the kernel of the second-order system that is formed through the multiplication of two first-order systems that are connected in parallel, having first-order Volterra kernels f and g, respectively [15]. The output of this system is obtained through the multiplication of the outputs of the two first-order systems:

$$y(t) = \int_{-\infty}^{\infty} f(\sigma_1)\, u(t-\sigma_1)\, d\sigma_1 \int_{-\infty}^{\infty} g(\sigma_2)\, u(t-\sigma_2)\, d\sigma_2$$

$$
\begin{aligned}
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\sigma_1)\, g(\sigma_2)\, u(t-\sigma_2) u(t-\sigma_1)\, d\sigma_1 d\sigma_2 \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_2(\sigma_1,\sigma_2)\, u(t-\sigma_2) u(t-\sigma_1)\, d\sigma_1 d\sigma_2
\end{aligned} \tag{2.59}
$$

where h_2 is the second-order kernel defined in equation (2.58). Clearly, the order of multiplication is unimportant in equation (2.59) and the output can also be written as

$$
\begin{aligned}
y(t) &= \int_{-\infty}^{\infty} g(\sigma_1)\, u(t-\sigma_1)\, d\sigma_1 \int_{-\infty}^{\infty} f(\sigma_2)\, u(t-\sigma_2)\, d\sigma_2 \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(\sigma_1)\, f(\sigma_2)\, u(t-\sigma_2) u(t-\sigma_1)\, d\sigma_1 d\sigma_2 \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_2(\sigma_2,\sigma_1)\, u(t-\sigma_2) u(t-\sigma_1)\, d\sigma_1 d\sigma_2
\end{aligned} \tag{2.60}
$$

where, as stated before, $h_2(\sigma_2,\sigma_1) = g(\sigma_1) f(\sigma_2)$ is not, in general, equal to $h_2(\sigma_1,\sigma_2)$. Hence, the second-order system has two different kernel representations, $h_2(\sigma_1,\sigma_2)$ and $h_2(\sigma_2,\sigma_1)$. Clearly, if equations (2.59) and (2.60) are added together and divided in half, we obtain

$$
y(t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_{2,sym}(\sigma_1,\sigma_2)\, u(t-\sigma_1) u(t-\sigma_2)\, d\sigma_1 d\sigma_2 \tag{2.61}
$$

where the symmetric kernel $h_{2,sym}(\sigma_1,\sigma_2)$ is defined as

$$
h_{2,sym}(\sigma_1,\sigma_2) = \frac{1}{2}\left\{h_2(\sigma_1,\sigma_2) + h_2(\sigma_2,\sigma_1)\right\} \tag{2.62}
$$

Using equation (2.62), a unique symmetric kernel can be derived for any second-order system. Higher-order symmetric kernels can be obtained using the following general expression:

$$
h_{n,sym}(\sigma_1,\ldots,\sigma_n) = \frac{1}{n!}\sum_{\pi(\cdot)} h_n\left(\sigma_{\pi(1)},\ldots,\sigma_{\pi(n)}\right) \tag{2.63}
$$

where $\pi(\cdot)$ represents any permutation of the integers 1 through n. In general, there are $n!$ possible permutations of the integers $1, \ldots, n$ and the resulting symmetric kernel satisfies

$$h_{n,sym}(\sigma_1, \ldots, \sigma_n) = h_{n,sym}\left(\sigma_{\pi(1)}, \ldots, \sigma_{\pi(n)}\right) \tag{2.64}$$

As in the second-order case, the symmetric forms of the higher-order kernels are unique for a given system. It is convenient to work with symmetric kernels because the order of the variables $\sigma_1, \ldots, \sigma_n$ is unimportant. Since all Volterra kernels can be put into a symmetric form using equation (2.63), one can always assume, without loss of generality, that the kernels are symmetric.

The triangular form of the nth-order Voltera kernel $h_{n,tri}$ satisfies the property

$$h_{n,tri}(\sigma_1, \sigma_2, \ldots, \sigma_n) = 0 \quad \text{for} \quad \sigma_{i+j} > \sigma_j \tag{2.65}$$

where i and j are positive integers. The resulting kernel is supported over a triangular domain. This domain is easy to visualize for the second-order kernel, where the kernel is supported in the triangular region bounded by $\sigma_1 = t$, $\sigma_2 = 0$, and $\sigma_1 = \sigma_2$. The condition in equation (2.65) can be explicitly incorporated into the expression for the nth-order kernel as follows:

$$h_{n,tri}(\sigma_1, \sigma_2, \ldots, \sigma_n) = h_{n,tri}(\sigma_1, \sigma_2, \ldots, \sigma_n)\, \delta_{-1}(\sigma_1 - \sigma_2) \cdots \delta_{-1}(\sigma_{n-1} - \sigma_n) \tag{2.66}$$

where δ_{-1} represents the unit step function, defined as

$$\delta_{-1}(\sigma_j - \sigma_{j+1}) := \begin{cases} 1 & \sigma_j \geq \sigma_{j+1} \\ 0 & \sigma_j < \sigma_{j+1} \end{cases} \tag{2.67}$$

It should be noted that equation (2.66) is not the only valid form for the triangular kernel. Indeed, any permutation of the arguments $\sigma_1, \ldots, \sigma_n$ in equation (2.66) also

yields a triangular kernel. Therefore, there are a total of $n!$ possible forms for an nth-order triangular kernel. As will be seen in the next section, triangular forms of the kernels naturally appear in Volterra series representations of ordinary differential equations.

Because every Volterra kernel can be expressed in a unique, symmetric form, it is not surprising that there is a relationship between the triangular and symmetric forms of a kernel. This relationship is given by

$$h_{n,tri}(\sigma_1, \sigma_2, \ldots, \sigma_n) = n!\, h_{n,sym}(\sigma_1, \sigma_2, \ldots, \sigma_n)\, \delta_{-1}(\sigma_1 - \sigma_2) \cdots \delta_{-1}(\sigma_{n-1} - \sigma_n) \tag{2.68}$$

The choice of which form of the kernel to use is a matter of convenience. In this dissertation, both the triangular and symmetric forms of the second-order kernel are considered. Multiwavelets that are easily adapted for the representation of symmetric kernels are derived and implemented. Alternatively, wavelets that are supported over triangular domains are derived and used to obtain low-order approximations of triangular second-order kernels.

Just as in the first-order case, higher-order Volterra kernels can also be analyzed in the frequency domain. The n-dimensional Fourier transform is defined as

$$H_n(j\omega_1, \ldots, j\omega_n) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} h_n(\sigma_1, \ldots, \sigma_n)\, e^{-j(\omega_1\sigma_1 + \cdots + \omega_n\sigma_n)} d\sigma_1 \ldots d\sigma_n \tag{2.69}$$

and the inverse Fourier transform takes the form

$$h_n(\sigma_1, \ldots, \sigma_n) = \frac{1}{(2\pi)^n} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} H_n(j\omega_1, \ldots, j\omega_n)\, e^{j(\omega_1\sigma_1 + \cdots + \omega_n\sigma_n)} d\omega_1 \ldots d\omega_n \tag{2.70}$$

As an example, the second-order Volterra kernel can be expressed in the frequency domain as

$$H_2(j\omega_1, j\omega_2) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h_2(\sigma_1, \sigma_n)\, e^{-j(\omega_1\sigma_1+\omega_2\sigma_2)} d\sigma_1 d\sigma_2 \tag{2.71}$$

where $H_2(j\omega_1, j\omega_2)$ is known as the kernel transform of the second-order kernel. Clearly, if $H_2(j\omega_1, j\omega_2)$ is known, the second-order kernel h_2 can be recovered using the inverse Fourier transform:

$$h_2(\sigma_1, \sigma_2) = \frac{1}{(2\pi)^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} H_2(j\omega_1, j\omega_2)\, e^{j(\omega_1\sigma_1+\omega_2\sigma_2)} d\omega_1 d\omega_2 \tag{2.72}$$

It should be noted that, just as in the first-order case, the existence of finite Fourier transforms of the higher-order kernels is dependent upon the condition

$$\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} |h_n(\sigma_1, \ldots, \sigma_n)|\, d\sigma_1 \cdots d\sigma_n < \infty \tag{2.73}$$

In Section 2.2, it was shown that, in the first-order case, this condition is both a necessary and sufficient condition for a linear system to be bounded-input, bounded-output (BIBO) stable. For higher-order, nonlinear systems, equation (2.73) is a sufficient condition for BIBO stability, but it has been shown by Schetzen [16] that it is not a necessary condition.

2.4 Volterra Series Representations of Ordinary Differential Equations

There is a distinct relationship between analytic ordinary differential equations (ODEs) and Volterra series. It can be shown that any ODE can be converted into an integral equation through the process of integration. The solution of the corresponding integral equation that is obtained using the method of successive substitutions, also known as Picard iteration, takes the form of a Volterra series. In this section, the relationship between ODEs and Volterra series will be discussed for several different time-invariant ODEs. Since an ODE of any given order can be expressed in terms

of a first-order state space equation, the discussion in this section is applicable to all analytic ODEs, regardless of their order.

Consider a linear, time-invariant ODE of the form

$$\begin{aligned} \dot{\underline{x}}(t) &= [A]\,\underline{x}(t) + \underline{b}u(t), \quad t \geq 0 \\ y(t) &= \underline{c}^T \underline{x}(t), \quad \underline{x}(0) = \underline{x}_0 \end{aligned} \tag{2.74}$$

where the input u is bounded and piecewise-continuous over the domain of interest. It is well known that the solution to equation (2.74) is given by

$$y(t) = \underline{c}^T e^{[A]t}\underline{x}_0 + \underline{c}^T \int_0^t e^{[A](t-\sigma)}\underline{b}u(\sigma)\,d\sigma \tag{2.75}$$

where the matrix exponential $e^{[A](t-\sigma)}$ represents the state transition matrix $[\Phi(t,\sigma)]$ for time-invariant systems. The matrix exponential is defined as

$$e^{[A]t} := \sum_{n=0}^{\infty} \frac{[A]^n (t)^n}{n!} = [I] + [A]\,t + \frac{1}{2!}[A]^2\ t^2 + \cdots \tag{2.76}$$

where $[I]$ is the identity matrix. It can be shown that, in general, the state transition matrix for both time-invariant and time-varying systems satisfies the following properties:

$$\begin{aligned} &(1) \qquad [\Phi(t,t)] = [I] \\ &(2) \qquad [\Phi(t,\sigma)][\Phi(\sigma,\tau)] = [\Phi(t,\tau)] \\ &(3) \qquad [\Phi(t,\sigma)]^{-1} = [\Phi(\sigma,t)] \end{aligned} \tag{2.77}$$

Clearly, equation (2.75) shows that the closed form solution of equation (2.74) takes the form of a first-order Volterra series,

$$y(t) = y_0(t) + \int_0^t h_1(t-\sigma)\, u(\sigma)\, d\sigma \tag{2.78}$$

where the zero-order term y_0 is given as

$$y_0(t) = \underline{c}^T e^{[A]t} \underline{x}_0 \tag{2.79}$$

and the first-order Volterra kernel takes the form

$$h_1(\sigma) = \underline{c}^T e^{[A]\sigma} \underline{b} \tag{2.80}$$

As expected, since equation (2.74) represents a linear system, the output is completely characterized in terms of the first-order Volterra operator and the zero-order term. Note that if the initial condition $\underline{x}_0$ is zero, the zero-order term vanishes. Equation (2.75) is typically obtained from the variation of constants formula. However, an alternative solution method, which generalizes to nonlinear equations, can be used to derive the same result. This procedure entails converting equation (2.74) into a Volterra integral equation and applying the method of successive substitutions. Equation (2.74) can be expressed as an integral equation via integration:

$$\underline{x}(t) = \underline{x}_0 + \int_0^t \{[A]\,\underline{x}(\sigma_1) + \underline{b}\, u(\sigma_1)\}\, d\sigma_1 \tag{2.81}$$

Equation (2.81) is a Volterra integral equation of the second kind which can be solved for the state vector $\underline{x}$. Note that the solution to equation (2.81) appears in the integrand, a characteristic of all integral equations. As demonstrated by Rugh [15], this equation can be solved using the method of successive substitutions, which assumes that $\underline{x}(\sigma_1)$ takes the form

$$\underline{x}(\sigma_1) = \underline{x}_0 + \int_0^{\sigma_1} \{[A]\,\underline{x}(\sigma_2) + \underline{b}\, u(\sigma_2)\}\, d\sigma_2 \tag{2.82}$$

Then, substituting into equation (2.81), we have

$$\begin{aligned}\underline{x}(t) &= \underline{x}_0 + \int_0^t \left\{ [A] \left\{ \underline{x}_0 + \int_0^{\sigma_1} \{[A]\,\underline{x}(\sigma_2) + \underline{b}u(\sigma_2)\}\, d\sigma_2 \right\} + \underline{b}u(\sigma_1) \right\} d\sigma_1 \\ &= \underline{x}_0 + [A]\, t\underline{x}_0 + \int_0^t \int_0^{\sigma_1} [A]\, \underline{b}u(\sigma_2)\, d\sigma_2 d\sigma_1 \\ &\quad + \int_0^t \underline{b}u(\sigma_1)\, d\sigma_1 + \int_0^t \int_0^{\sigma_1} [A]^2\, \underline{x}(\sigma_2)\, d\sigma_2 d\sigma_1 \end{aligned} \tag{2.83}$$

Using equation (2.82), $\underline{x}(\sigma_2)$ can be expressed as

$$\underline{x}(\sigma_2) = \underline{x}_0 + \int_0^{\sigma_2} \{[A]\,\underline{x}(\sigma_3) + \underline{b}\, u(\sigma_3)\}\, d\sigma_3$$

and substituted into equation (2.83) to obtain

$$\begin{aligned}\underline{x}(t) &= \underline{x}_0 + [A]\, t\underline{x}_0 + \int_0^t \int_0^{\sigma_1} [A]\, \underline{b}\, u(\sigma_2)\, d\sigma_2 d\sigma_1 + \int_0^t \underline{b}\, u(\sigma_1)\, d\sigma_1 \\ &\quad + \int_0^t \int_0^{\sigma_1} [A]^2 \left\{ \underline{x}_0 + \int_0^{\sigma_2} \{[A]\,\underline{x}(\sigma_3) + \underline{b}\, u(\sigma_3)\}\, d\sigma_3 \right\} d\sigma_2 d\sigma_1 \\ &= \left\{ [I] + [A]\, t + \tfrac{1}{2} [A]^2 t^2 \right\} \underline{x}_0 + \int_0^t \underline{b}\, u(\sigma_1)\, d\sigma_1 \\ &\quad + \int_0^t \int_0^{\sigma_1} [A]\, \underline{b}\, u(\sigma_2)\, d\sigma_2 d\sigma_1 + \int_0^t \int_0^{\sigma_1} \int_0^{\sigma_2} [A]^2\, \underline{b}u(\sigma_3) d\sigma_3 d\sigma_2 d\sigma_1 \\ &\quad + \int_0^t \int_0^{\sigma_1} \int_0^{\sigma_2} [A]^3\, \underline{x}(\sigma_3) d\sigma_3 d\sigma_2 d\sigma_1 \end{aligned} \tag{2.84}$$

Note that the third and fourth integrals in equation (2.84) can be simplified by changing the order of integration. For example, the third integral can be rewritten as

$$\begin{aligned}\int_0^t \int_0^{\sigma_1} [A]\, \underline{b}\, u(\sigma_2)\, d\sigma_2 d\sigma_1 &= \int_0^t \int_{\sigma_2}^t [A]\, \underline{b}\, u(\sigma_2)\, d\sigma_1 d\sigma_2 \\ &= \int_0^t [A]\, (t - \sigma_2)\, \underline{b}\, u(\sigma_2)\, d\sigma_2 \end{aligned} \tag{2.85}$$

Similarly, the fourth integral can be simplified to the form

$$\int_0^t \int_0^{\sigma_1} \int_0^{\sigma_2} [A]^2 \underline{b}\, u(\sigma_3) d\sigma_3 d\sigma_2 d\sigma_1 = \int_0^t [A]^2 (t-\sigma_3)^2 \underline{b}\, u(\sigma_3)\, d\sigma_3 \tag{2.86}$$

Using these results, equation (2.84) can be written as

$$\begin{aligned} \underline{x}(t) &= \left\{[I] + [A]\, t + \tfrac{1}{2}[A]^2 t^2\right\} \underline{x}_0 \\ &\quad + \int_0^t \left\{[I] + [A](t-\sigma_1) + \tfrac{1}{2}[A]^2 (t-\sigma_1)^2\right\} \underline{b} u(\sigma_1)\, d\sigma_1 \\ &\quad + \int_0^t \int_0^{\sigma_1} \int_0^{\sigma_2} [A]^3 \underline{x}(\sigma_3) d\sigma_3 d\sigma_2 d\sigma_1 \end{aligned} \tag{2.87}$$

It can be demonstrated that, as the number of iterations in the method of successive substitutions approaches infinity, the last term (the term involving the state vector) approaches zero [15]. Using this fact and the definition of the matrix exponential given in equation (2.76), as the number of iterations approaches infinity, we obtain

$$\underline{x}(t) = e^{[A]t} \underline{x}_0 + \int_0^t e^{[A](t-\sigma_1)} \underline{b}\, u(\sigma_1)\, d\sigma_1$$

and, finally,

$$y(t) = \underline{c}^T \underline{x}(t) = \underline{c}^T e^{[A]t} \underline{x}_0 + \underline{c}^T \int_0^t e^{[A](t-\sigma_1)} \underline{b}\, u(\sigma_1)\, d\sigma_1 \tag{2.88}$$

This result is identical to the solution given by equation (2.75).

It has been shown that, by transforming an ordinary differential equation into an integral equation and applying the method of successive substitutions, a Volterra series representation of the solution can be obtained. In the previous example, this procedure was applied to a linear system, and the resulting series consisted only of the zero-order term and the first-order Volterra operator. Volterra series representations of nonlinear systems include higher-order terms and are generally infinite series. Consider a bilinear system of the form

$$\begin{aligned} \dot{\underline{x}}(t) &= [A]\,\underline{x}(t) + [D]\,\underline{x}(t)\,u(t) + \underline{b}\,u(t), \quad t \geq 0 \\ y(t) &= \underline{c}^T \underline{x}(t), \qquad \underline{x}(0) = \underline{x}_0 \end{aligned} \tag{2.89}$$

where the input u is bounded and piecewise-continuous over the domain of interest. Rugh [15] showed that the evaluation of equation (2.89) can be simplified via a change of variables:

$$\underline{z}(t) = e^{-[A]t}\underline{x}(t) \quad \Longrightarrow \quad \underline{x}(t) = e^{[A]t}\underline{z}(t) \tag{2.90}$$

Note that

$$\begin{aligned} \frac{d}{dt}e^{[A]t} &= \frac{d}{dt}\left\{[I] + [A]\,t + \tfrac{1}{2}[A]^2 t^2 + \tfrac{1}{3}[A]^3 t^3 + \cdots\right\} \\ &= [A] + [A]^2 t + \tfrac{1}{2}[A]^3 t^3 + \cdots \\ &= [A]\left\{[I] + [A]\,t + \tfrac{1}{2}[A]^2 t^2 + \cdots\right\} \\ &= [A]\,e^{[A]t} \end{aligned}$$

Then, differentiating equation (2.90), we have

$$\dot{\underline{x}}(t) = [A]\,e^{[A]t}\underline{z}(t) + e^{[A]t}\dot{\underline{z}}(t) \tag{2.91}$$

Substituting for $\underline{x}(t)$ and $\dot{\underline{x}}(t)$ in equation (2.89), we obtain

$$[A]\,e^{[A]t}\underline{z}(t) + e^{[A]t}\dot{\underline{z}}(t) = [A]\,e^{[A]t}\underline{z}(t) + [D]\,e^{[A]t}\underline{z}(t)\,u(t) + \underline{b}\,u(t)$$

which can be simplified to give the following equation for the new variable $\underline{z}$:

$$\dot{\underline{z}}(t) = e^{-[A]t}\,[D]\,e^{[A]t}\underline{z}(t)\,u(t) + e^{-[A]t}\underline{b}\,u(t) \tag{2.92}$$

Equation (2.92) can be integrated to obtain the integral equation

$$\underline{z}(t) = \underline{z}_0 + \int_0^t e^{-[A]\sigma_1}[D]\, e^{[A]\sigma_1}\underline{z}(\sigma_1)\, u(\sigma_1) d\sigma_1 + \int_0^t e^{-[A]\sigma_1}\underline{b}\, u(\sigma_1)\, d\sigma_1 \qquad (2.93)$$

Applying the method of successive substitutions to solve this integral equation, it is assumed that $\underline{z}(\sigma_1)$ takes the form

$$\underline{z}(\sigma_1) = \underline{z}_0 + \int_0^{\sigma_1} e^{-[A]\sigma_2}[D]\, e^{[A]\sigma_2}\underline{z}(\sigma_2)\, u(\sigma_2) d\sigma_2 + \int_0^{\sigma_1} e^{-[A]\sigma_2}\underline{b}\, u(\sigma_2)\, d\sigma_2$$

Then, substituting into equation (2.93), we obtain

$$\begin{aligned}\underline{z}(t) &= \underline{z}_0 + \int_0^t e^{-[A]\sigma_1}[D]\, e^{[A]\sigma_1}\underline{z}_0 u(\sigma_1)\, d\sigma_1 + \int_0^t e^{-[A]\sigma_1}\underline{b}u(\sigma_1)\, d\sigma_1 \\ &+ \int_0^t\int_0^{\sigma_1} e^{-[A]\sigma_1}[D]\, e^{[A]\sigma_1}e^{-[A]\sigma_2}\underline{b}\, u(\sigma_2)\, u(\sigma_1)\, d\sigma_2 d\sigma_1 \\ &+ \int_0^t\int_0^{\sigma_1} e^{-[A]\sigma_1}[D]\, e^{[A]\sigma_1}e^{-[A]\sigma_2}[D]\, e^{[A]\sigma_2}\underline{z}(\sigma_2)\, u(\sigma_2)\, u(\sigma_1)\, d\sigma_2 d\sigma_1\end{aligned} \qquad (2.94)$$

Note that, from property (3) of the state transition matrix,

$$e^{[A]\sigma_1}e^{-[A]\sigma_2} = [\Phi(\sigma_1, 0)][\Phi(0, \sigma_2)] = [\Phi(\sigma_1, \sigma_2)] = e^{[A](\sigma_1 - \sigma_2)}$$

Then, equation (2.94) can be simplified to the form

$$\begin{aligned}\underline{z}(t) &= \underline{z}_0 + \int_0^t e^{-[A]\sigma_1}[D]\, e^{[A]\sigma_1}\underline{z}_0 u(\sigma_1)\, d\sigma_1 + \int_0^t e^{-[A]\sigma_1}\underline{b}u(\sigma_1)\, d\sigma_1 \\ &+ \int_0^t\int_0^{\sigma_1} e^{-[A]\sigma_1}[D]\, e^{[A](\sigma_1-\sigma_2)}\underline{b}u(\sigma_2)\, u(\sigma_1)\, d\sigma_2 d\sigma_1 \\ &+ \int_0^t\int_0^{\sigma_1} e^{-[A]\sigma_1}[D]\, e^{[A](\sigma_1-\sigma_2)}[D]\, e^{[A]\sigma_2}\underline{z}(\sigma_2)\, u(\sigma_2)\, u(\sigma_1)\, d\sigma_2 d\sigma_1\end{aligned} \qquad (2.95)$$

Continuing with the successive substitutions, $\underline{z}(\sigma_2)$ is assumed to take the form

$$\underline{z}\,(\sigma_2) = \underline{z}_0 + \int_0^{\sigma_2} e^{-[A]\sigma_3}\,[D]\,e^{[A]\sigma_3}\underline{z}\,(\sigma_3)\,u\,(\sigma_3)d\sigma_3 + \int_0^{\sigma_2} e^{-[A]\sigma_3}\underline{b}\;u\,(\sigma_3)\,d\sigma_3$$

Then, we have the following expression for $\underline{z}$:

$$\begin{aligned}
\underline{z}\,(t) &= \underline{z}_0 + \int_0^t e^{-[A]\sigma_1}\,[D]\,e^{[A]\sigma_1}\underline{z}_0 u\,(\sigma_1)\,d\sigma_1 + \int_0^t e^{-[A]\sigma_1}\underline{b}\;u\,(\sigma_1)\,d\sigma_1 \\
&\quad + \int_0^t \int_0^{\sigma_1} e^{-[A]\sigma_1}\,[D]\,e^{[A](\sigma_1-\sigma_2)}\,[D]\,e^{[A]\sigma_2}\underline{z}_0 u\,(\sigma_2)\,u\,(\sigma_1)\,d\sigma_2 d\sigma_1 \\
&\quad + \int_0^t \int_0^{\sigma_1} e^{-[A]\sigma_1}\,[D]\,e^{[A](\sigma_1-\sigma_2)}\underline{b}\;u\,(\sigma_2)\,u\,(\sigma_1)\,d\sigma_2 d\sigma_1 \\
&\quad + \int_0^t \int_0^{\sigma_1} \int_0^{\sigma_2} e^{-[A]\sigma_1}\,[D]\,e^{[A](\sigma_1-\sigma_2)}\,[D]\,e^{[A](\sigma_2-\sigma_3)}\underline{b} \\
&\qquad\qquad u\,(\sigma_3)\,u\,(\sigma_2)\,u\,(\sigma_1)\,d\sigma_3 d\sigma_2 d\sigma_1 \\
&\quad + \int_0^t \int_0^{\sigma_1} \int_0^{\sigma_2} e^{-[A]\sigma_1}\,[D]\,e^{[A](\sigma_1-\sigma_2)}\,[D]\,e^{[A](\sigma_2-\sigma_3)}\,[D]\,e^{[A]\sigma_3} \\
&\qquad\qquad \underline{z}\,(\sigma_3)\,u\,(\sigma_3)\,u\,(\sigma_2)\,u\,(\sigma_1)\,d\sigma_3 d\sigma_2 d\sigma_1
\end{aligned} \tag{2.96}$$

Just as in the previous example, as the number of iterations in the method of successive substitutions is increased, the last term approaches zero. Using equation (2.90) to transform the state variable back to $\underline{x}$, and using the output equation given in equation (2.89), we obtain the following expression for the output of the bilinear state equation:

$$\begin{aligned}
y\,(t) &= \underline{c}^T e^{[A]t}\underline{x}_0 + \underline{c}^T \int_0^t e^{[A](t-\sigma_1)}\,[D]\,e^{[A]\sigma_1}\underline{x}_0 u\,(\sigma_1)\,d\sigma_1 \\
&\quad + \underline{c}^T \int_0^t e^{[A](t-\sigma_1)}\underline{b} u\,(\sigma_1)\,d\sigma_1 \\
&\quad + \underline{c}^T \sum_{k=2}^{\infty} \int_0^t \int_0^{\sigma_1} \cdots \int_0^{\sigma_{k-1}} e^{[A](t-\sigma_1)}\,[D]\,e^{[A](\sigma_1-\sigma_2)} \cdots \\
&\qquad\qquad [D]\,e^{[A]\sigma_k}\underline{x}_0 u\,(\sigma_k) \ldots u\,(\sigma_1)\,d\sigma_k \ldots d\sigma_1
\end{aligned} \tag{2.97}$$

$$+\underline{c}^T \sum_{k=2}^{\infty} \int_0^t \int_0^{\sigma_1} \cdots \int_0^{\sigma_{k-1}} e^{[A](t-\sigma_1)}\,[D]\,e^{[A](\sigma_1-\sigma_2)} \cdots$$
$$[D]\,e^{[A](\sigma_{k-1}-\sigma_k)}\underline{b}u\,(\sigma_k) \ldots u\,(\sigma_1)\,d\sigma_k \ldots d\sigma_1$$

Equation (2.97) is a rather complicated Volterra series representation of the output of the bilinear system. If the initial condition $\underline{x}_0$ is zero, equation (2.97) takes a much simpler form. Rugh [15] has shown that, through an appropriate change of variables, a given state equation with nonzero initial conditions can be transformed into one with zero initial conditions. If the initial conditions are zero, equation (2.97) reduces to

$$\begin{aligned} y\,(t) &= \underline{c}^T \int_0^t e^{[A](t-\sigma_1)}\underline{b}\,u\,(\sigma_1)\,d\sigma_1 \qquad (2.98)\\ &+\underline{c}^T \sum_{k=2}^{\infty} \int_0^t \int_0^{\sigma_1} \cdots \int_0^{\sigma_{k-1}} e^{[A](t-\sigma_1)}\,[D]\,e^{[A](\sigma_1-\sigma_2)} \cdots \\ &\qquad [D]\,e^{[A](\sigma_{k-1}-\sigma_k)}\underline{b}\,u\,(\sigma_k) \cdots u\,(\sigma_1)\,d\sigma_k \cdots d\sigma_1 \end{aligned}$$

It is not difficult to show that equation (2.98) is, indeed, in the form of a Volterra series. Recall that a Volterra series representation of a nonlinear system can be written as

$$\begin{aligned} y\,(t) &= \int_0^t h_1\,(t-\sigma_1)\,u\,(\sigma_1)\,d\sigma_1 \qquad (2.99)\\ &+\int_0^t \int_0^t h_2\,(t-\sigma_1, t-\sigma_2)\,u\,(\sigma_2)\,u\,(\sigma_1)\,d\sigma_2 d\sigma_1 + \cdots \end{aligned}$$

Note that, from the properties of the state transition matrix,

$$e^{[A](\sigma_{k-1}-\sigma_k)} = [\Phi\,(\sigma_{k-1}, t)]\,[\Phi\,(t, \sigma_k)] = e^{-[A](t-\sigma_{k-1})}e^{[A](t-\sigma_k)}$$

Then, equation (2.98) can be written as

$$
\begin{aligned}
y(t) &= \int_0^t \underline{c}^T e^{[A](t-\sigma_1)} \underline{b} u(\sigma_1)\, d\sigma_1 \\
&\quad + \int_0^t \int_0^{\sigma_1} \underline{c}^T e^{[A](t-\sigma_1)} [D]\, e^{-[A](t-\sigma_1)} e^{[A](t-\sigma_2)} \underline{b} u(\sigma_2) u(\sigma_1)\, d\sigma_2 d\sigma_1 + \cdots
\end{aligned} \tag{2.100}
$$

Clearly, equation (2.100) is of the same form as the Volterra series given in equation (2.99), where the first and second-order Volterra kernels are given by

$$
h_1(\sigma_1) = \underline{c}^T e^{[A]\sigma_1} \underline{b} \tag{2.101}
$$

$$
\begin{aligned}
h_2(\sigma_1, \sigma_2) &= \underline{c}^T e^{[A]\sigma_1} [D]\, e^{-[A]\sigma_1} e^{[A]\sigma_2} \underline{b} \\
&= \underline{c}^T e^{[A]\sigma_1} [D] e^{[A](\sigma_2 - \sigma_1)} \underline{b}
\end{aligned} \tag{2.102}
$$

The same technique that was used to obtain Volterra series representations of the output of the linear and bilinear state equations can be used to generate Volterra series for general time-invariant, nonlinear ODEs of the form

$$
\begin{aligned}
\dot{\underline{x}}(t) &= \underline{f}(\underline{x}(t), u(t)), \quad t \geq 0 \\
y(t) &= h(\underline{x}(t)), \qquad \underline{x}(0) = \underline{x}_0
\end{aligned} \tag{2.103}
$$

It is interesting to note that ODEs naturally give rise to Volterra series representations that are composed of kernels that are in the triangular form described in the previous section. It should also be noted that the method of successive substitutions is not the only technique available for deriving analytical kernels from ODEs. Ku and Wolf [9] used a Taylor series expansion of the nonlinear terms in ODEs to derive analytical forms of the kernels. Fliess *et al.* [12] used iterated integrals to derive a

general expression for the Volterra kernels corresponding to nonlinear ODEs. As a final example, Schetzen [16] introduced the concept of the pth-order inverse in order to invert the pth-order Volterra operator and determine the corresponding kernel.

2.5 Existence, Uniqueness, and Convergence Issues

A review of Volterra series would be incomplete without some discussion of the existence, uniqueness, and convergence of these representations. It is important to distinguish which types of systems are amenable to Volterra series representations and whether or not such representations are unique. Furthermore, it is essential to consider the convergence properties of Volterra series in order to apply them to the analysis of dynamical systems. These fundamental issues are addressed in this section.

Boyd and Chua [8] established that a system must possess fading memory in order to be represented in terms of a Volterra series. While the concept of fading memory was alluded to in the early works of Volterra [7] and Wiener [13], Boyd and Chua were the first to provide a formal definition of the term.

Definition 2.1 [8]: An operator N has *fading memory* on a subset K of $C(\mathbb{R})$, the set of all bounded continuous real-valued functions, if there is a decreasing function $w : \mathbb{R} \rightarrow (0, 1]$, $\lim_{t \rightarrow \infty} w(t) = 0$, such that for each $u \in K$ and $\epsilon > 0$, there is a $\delta > 0$ such that for all $v \in K$,

$$\sup_{t \leq 0} |u(t) - v(t)| w(-t) < \delta \Longrightarrow |Nu(0) - Nv(0)| < \epsilon \tag{2.104}$$

In this definition, time $t = 0$ corresponds to the present time. In words, Definition 2.1 states that a system exhibits fading memory if two inputs signals that are close in the recent past, but not necessarily close in the remote past, result in close present outputs. The weighting function $w(-t)$ approaches 0 as the time approaches $-\infty$ to

indicate the fading influence of past inputs. For a fading memory system, an input occurring in the distant past will eventually have no influence on the present output. Boyd and Chua [8] noted that fading memory corresponds to a stronger requirement than continuity, which states that input signals that are close over all past time result in close present outputs. They also remarked that for linear time-invariant systems, fading memory is equivalent to having a convolution representation.

Originally, approximation results for Volterra series representations of time-invariant systems were restricted by the limitations of the Stone-Weierstrass theorem. In short, these limitations restricted the approximation results to input signals defined on a finite time interval and applied to the approximation of an operator N over a finite time interval. The fading memory concept enabled Boyd and Chua [8] to state an approximation theorem that was free of these constraints:

Theorem 2.1 [8]: Let $\epsilon > 0$ and

$$K := \{u \in C(\mathbb{R}) : \ |u(t)| \leq M_1, \ |u(s) - u(t)| \leq M_2(s-t) \text{ for } t \leq s\} \tag{2.105}$$

Suppose that N is any time-invariant operator with fading memory on K. Then, there is a finite Volterra series operator $\hat{N}$ such that, for all $u \in K$,

$$\|Nu - \hat{N}u\|_\infty \leq \epsilon \tag{2.106}$$

Note that the ∞-norm of a function f is defined as

$$\|f\|_\infty := \sup_{t \in \mathbb{R}} |f(t)| \tag{2.107}$$

Theorem 2.1 basically states that a time-invariant operator with fading memory, subject to a bounded input excitation, can be approximated to arbitrary accuracy in

terms of a truncated Volterra series representation. By invoking the requirement of fading memory, the theorem is not subject to any of the previous restrictions.

Boyd *et al.* [10] addressed the issue of uniqueness of Volterra series representations. They provided the following uniqueness theorem, which is stated here without proof:

Theorem 2.2 [10]: Suppose N and M are Volterra series operators with kernels h_n and g_n, respectively. Then $N = M$ if and only if $h_{n,sym} = g_{n,sym}$ for all n.

Theorem 2.2 states that two Volterra series representations are equivalent if and only if the corresponding symmetric kernels are equivalent. As noted earlier, although Volterra kernels can be expressed in different forms, the symmetric form of the kernels is always unique. Therefore, Theorem 2.2 assures that the set of Volterra kernels that characterize a given system are unique to that system.

The convergence of Volterra series was addressed in the following theorem from Ku and Wolf [9]:

Theorem 2.3 [9]: Let

$$y_n(t) = \int_0^\infty \cdots \int_0^\infty h_n(\sigma_1, \ldots, \sigma_n) u(t-\sigma_1) \cdots u(t-\sigma_n) d\sigma_1 \cdots d\sigma_n \tag{2.108}$$

Then, for a bounded input function u that satisfies

$$|u(t)| \leq M \quad \forall\, t \in \mathbb{R} \tag{2.109}$$

for some constant $M > 0$, we have

$$\left| \sum_{n=1}^{\infty} y_n(t) \right| \leq \sum_{n=1}^{\infty} a_n M^n \tag{2.110}$$

where

$$a_n = \int_0^\infty \cdots \int_0^\infty |h_n(\sigma_1, \ldots, \sigma_n)| d\sigma_1 \cdots d\sigma_n \tag{2.111}$$

The proof of Theorem 2.3 follows a similar format as the proofs of the stability results given earlier in this chapter. Theorem 2.3 implies that, similar to Taylor series expansions, a radius of convergence for Volterra series can be defined as [9]

$$\rho = \left(\limsup_{n \to \infty} |a_n|^{1/n} \right)^{-1} \tag{2.112}$$

Then, absolute convergence of the Volterra series is guaranteed for any input that satisfies

$$\|u\|_\infty < \rho \tag{2.113}$$

Boyd *et al.* [10] presented a slightly different form of Theorem 2.3 that discussed convergence of the Volterra series in terms of a gain bound function f, defined as

$$f(x) := \sum_{n=1}^{\infty} \|h_n\|_\infty \, x^n \tag{2.114}$$

As noted by Boyd *et al.* [10], Theorem 2.3 can be used to compute an error bound for a truncated Volterra series. Denoting H as the infinite Volterra series and $H^{(k)}$ as the truncated Volterra series

$$H^{(k)}u(t) = \sum_{n=1}^{k} \int_0^\infty \cdots \int_0^\infty h_n(\sigma_1, \ldots, \sigma_n) u(t - \sigma_1) \cdots u(t - \sigma_n) d\sigma_1 \cdots d\sigma_n \tag{2.115}$$

the truncation error is bounded as

$$\|Hu - H^{(k)}u\| \leq \sum_{n=k+1}^{\infty} a_n \|u\|_\infty^n \tag{2.116}$$

It then follows that the truncation error is $o(\|u\|^k)$ [10]. As a final note, Sandberg [11] has addressed the convergence properties of the "doubly finite" Volterra series in which, in addition to truncating the series, the kernels are also truncated after a finite period of time. This assumption is frequently employed in practice and is consistent with the concept of fading memory, which dictates that the Volterra kernels of a fading memory system should decay to zero in a finite period of time. Sandberg has also investigated the convergence of discrete-time Volterra series representations [97] and the convergence of Volterra series for time-varying nonlinear systems [98].

CHAPTER 3
REVIEW OF WAVELETS

This chapter presents an overview of wavelets and multiresolution analysis. Because this field is relatively new, this chapter is intended to be a self-contained review of some of the basic properties of wavelets, as well as how they are used in the approximation of functions and signals. In particular, the decomposition formulas that are implemented in the calculation of the discrete wavelet transform are derived for a number of different wavelet bases. Section 3.1 gives an introduction to wavelets and multiresolution analysis. Then, in Section 3.2, wavelets are compared to two commonly used tools for the approximation of functions, Fourier analysis and the finite element method. In Section 3.3, the Haar wavelet is introduced as a prototype wavelet for which many general wavelet properties are easy to visualize. In Section 3.4, the discussion is extended to include all orthonormal wavelet families. Then, continuing in order of increasing generality, orthonormal multiwavelets, wavelets that are generated from multiple scaling functions, are considered in Section 3.5. Finally, in Section 3.6, tensor product wavelets are introduced as a means of approximating two-dimensional and higher-dimensional functions. The material in this chapter forms a large portion of the background that will be useful later in deriving wavelet-based representations of first and second-order Volterra kernels.

3.1 Wavelets and Multiresolution Analysis

In simple terms, a wavelet is a compactly-supported, oscillatory function that is typically constructed to satisfy additional properties such as smoothness, orthogonality, and regularity conditions. A wavelet basis is composed of the scaled translates and dilates of the wavelet ψ, known as the "mother wavelet", and forms a basis for

the Hilbert space $L^2(\mathbb{R})$. The space $L^2(\mathbb{R})$ contains all square-integrable functions, or functions that satisfy

$$\int_{\mathbb{R}} |f(x)|^2 dx < \infty \tag{3.1}$$

This space has an associated inner product defined as

$$\langle f, g \rangle_{L^2(\mathbb{R})} := \int_{\mathbb{R}} f(x)\overline{g(x)}dx \tag{3.2}$$

where the overbar denotes complex conjugation. Every wavelet ψ has an associated scaling function ϕ. The scaling function and wavelet pair (ϕ, ψ) are derived from the solution of the two-scale equations:

$$\phi(x) = \sqrt{2}\sum_s a_s \phi(2x - s) \tag{3.3}$$

$$\psi(x) = \sqrt{2}\sum_s b_s \phi(2x - s) \tag{3.4}$$

where $\{a_s\}$ and $\{b_s\}$ are constant scaling function and wavelet filters. These equations state that the scaling function ϕ and the wavelet ψ are formed as linear combinations of scaling functions that have been contracted to double the frequency (or half the support). Because the scaling functions are used to form wavelets, they are also known as generators.

By construction, the scaling functions form a multiresolution analysis for the analysis of functions and signals in $L^2(\mathbb{R})$. Conceptually, a multiresolution analysis decomposes a function into components that show varying amounts of detail, similar to the effect achieved when one views an object through binoculars and varies the magnification power. More rigorously, a sequence of subspaces $\{V_j\}_{j\in\mathbb{Z}} \subset L^2(\mathbb{R})$, where $\mathbb{Z}$ is the set of all integers, forms a multiresolution analysis provided that

(1) $\cdots V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \cdots$

(2) $\bigcap_{j \in \mathbb{Z}} V_j = \{0\}$

(3) $\overline{\bigcup_{j \in \mathbb{Z}} V_j} = L^2(\mathbb{R})$ (3.5)

(4) $f(x) \in V_j \iff f(2x) \in V_{j+1}$

(5) $f(x) \in V_j \implies f(x-k) \in V_j \quad \forall\, k \in \mathbb{Z}$

(6) The family of integer translates $\{\phi(x-k)\}_{k \in \mathbb{Z}}$ forms a Riesz basis for V_0

A multiresolution analysis is obtained by forming a series of nested spaces $V_{j-1} \subset V_j$, $j \in \mathbb{Z}$. These spaces are generated from the scaled translates and dilates of the scaling function ϕ. Later in this chapter, the more general case in which the spaces $\{V_j\}$ are generated from a set of r scaling functions $\{\phi^1, \ldots, \phi^r\}$ will be discussed. For the single-generator case, the space V_0 is defined as the span of integer translates of ϕ:

$$V_0 := span\,\{\phi(\cdot - k)\}_{k \in \mathbb{Z}} \tag{3.6}$$

In general, the space V_j is defined as

$$V_j := span\,\{\phi_{j,k}\}_{k \in \mathbb{Z}} \tag{3.7}$$

where each function $\phi_{j,k}$ is defined as

$$\phi_{j,k}(x) := 2^{j/2}\phi\left(2^j x - k\right) \tag{3.8}$$

From equation (3.8), it is clear that the functions $\{\phi_{j,k}\}$ are formed as scaled dilates and translates of the scaling function ϕ. The integer j is known as the dilation index and the integer k represents the translation index. The $2^{j/2}$ term is a normalization

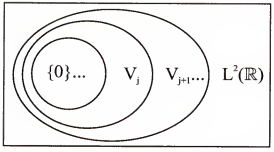


Figure 3.1: Nested Spaces $\{V_j\} \subset L^2(\mathbb{R})$

factor that ensures that the $L^2(\mathbb{R})$ norm of each scaling function is one. The $L^2(\mathbb{R})$ norm of a function f is given by

$$\|f\|_{L^2(\mathbb{R})} := \left[\langle f, f\rangle_{L^2(\mathbb{R})}\right]^{\frac{1}{2}} = \left[\int_{\mathbb{R}} |f(x)|^2 dx\right]^{\frac{1}{2}} \tag{3.9}$$

It is clear that, as the index j is increased, the scaling functions that span the corresponding space V_j are supported on increasingly finer grids. These grids are dyadic in nature, since increasing j by one gives a grid that has twice the resolution. As $j \to \infty$, the scaling functions are supported on an infinitesimal grid. In the limit, any function in $L^2(\mathbb{R})$ can be represented exactly. Thus, as $j \to \infty$ the spaces $\{V_j\}$ approach $L^2(\mathbb{R})$. Conversely, as j is decreased, the spaces $\{V_j\}$ are spanned by scaling functions that have increasingly wider support. As $j \to -\infty$, the spaces $\{V_j\}$ approach $\{0\}$, the space containing only the zero function. The reason for this is that, as $j \to -\infty$, the scaling functions are supported on an infinite domain. Thus, to this point, it is clear that the spaces $\{V_j\}$ satisfy the first three multiresolution conditions. These conditions are summarized in Figure (3.1).

By construction, the fourth and fifth requirements for a multiresolution analysis are satisfied. If a function $f(x)$ is contained in the space V_j, the function $f(2x)$ must be contained in the space V_{j+1}. This is apparent because the space V_{j+1} is spanned by scaling functions with double the frequency as those that span V_j. The fifth property states that if f is contained in V_j, the integer translates of f are also contained in

V_j. This is true because the space V_j is spanned by all integer translates of the dilated scaling function $\phi(2^j x)$. The final requirement for a multiresolution analysis is that the family of integer translates $\{\phi(x-k)\}_{k\in\mathbb{Z}}$ forms a Riesz basis for V_0. This requirement is satisfied if the family of translates $\{\phi(x-k)\}$ span V_0 and there exist constants $0 < A \leq B < \infty$ such that, for all sequences $\{p_k\}_{k\in\mathbb{Z}} \in L^2(\mathbb{Z})$,

$$A \sum_k |p_k|^2 \leq \left\| \sum_k p_k \phi(x-k) \right\|^2 \leq B \sum_k |p_k|^2 \tag{3.10}$$

(see Ref. [75]). It will be noted here, without proof, that scaling functions satisfy equation (3.10), commonly known as the frame condition. In particular, if the family of translates $\{\phi(x-k)\}$ forms an orthonormal basis for V_0, then equation (3.10) becomes an equality with $A = B = 1$ [99].

At this point, the relationship of the wavelet ψ to a multiresolution analysis has not yet been clarified. By construction, wavelets span the orthogonal complement spaces $\{W_j\}$ defined as the differences between two consecutive spaces $\{V_j\}$:

$$W_j := V_{j+1} \ominus V_j \tag{3.11}$$

The wavelet space W_j is defined as

$$W_j := span\{\psi_{j,k}\}_{k\in\mathbb{Z}} \tag{3.12}$$

The notation $\psi_{j,k}$ represents the scaled translates and dilates of the wavelet ψ:

$$\psi_{j,k}(x) := 2^{j/2}\psi\left(2^j x - k\right) \tag{3.13}$$

From equation (3.11), it is apparent that a given wavelet space W_j provides the details that are present in a representation of a function in V_{j+1}, but are not present in the coarser representation of the function in V_j. In practice, a fine-scale space V_j can be decomposed as

$$V_j = V_{j-1} \oplus W_{j-1} \tag{3.14}$$

Clearly, by extension, the space V_{j-1} can be decomposed as

$$V_{j-1} = V_{j-2} \oplus W_{j-2} \tag{3.15}$$

and equation (3.14) becomes

$$V_j = V_{j-2} \oplus W_{j-2} \oplus W_{j-1} \tag{3.16}$$

Through a recursive application of equation (3.14), the following multilevel decomposition of V_j is obtained:

$$V_j = V_{j_0} \oplus W_{j_0} \oplus W_{j_0+1} \oplus \cdots \oplus W_{j-2} \oplus W_{j-1} \tag{3.17}$$

where j_0 represents the coarsest level used in the decomposition. Therefore, the space V_j is equivalent to a sum of wavelet spaces corresponding to a number of different resolution levels plus a coarse-scale space V_{j_0}. In the limit as $j_0 \to -\infty$, the decomposition of V_j takes the form

$$V_j = \bigoplus_{l=-\infty}^{j-1} W_l \tag{3.18}$$

Furthermore, as $j \to \infty$, $V_j \to L^2(\mathbb{R})$. In that case, we have

$$L^2(\mathbb{R}) = \bigoplus_{l=-\infty}^{\infty} W_l \tag{3.19}$$

Therefore, the wavelets that collectively span the spaces $\{W_j\}_{j\in\mathbb{Z}}$ form a basis for $L^2(\mathbb{R})$.

In practice, a wavelet-based multiresolution analysis of a function f is obtained by first forming a single-scale approximation of the function in terms of the scaling functions that span V_j. This single-scale approximation takes the form

$$f_j(x) = \sum_k \alpha_{j,k}\phi_{j,k}(x) \tag{3.20}$$

where $\{\alpha_{j,k}\}$ represent constant coefficients in the expansion. The notation f_j denotes an approximation of f in V_j. The calculation of the coefficients $\{\alpha_{j,k}\}$ is particularly straightforward if the scaling functions $\{\phi_{j,k}\}$ are mutually orthonormal. The scaling functions are mutually orthonormal if

$$\int_{\mathbb{R}} \phi_{j,k}(x)\phi_{j,m}(x)dx = \delta_{k,m} \tag{3.21}$$

where $\delta_{k,m}$ denotes the Kronecker delta, defined as

$$\delta_{k,m} := \begin{cases} 0 & k \neq m \\ 1 & k = m \end{cases} \tag{3.22}$$

In this case, the coefficients $\{\alpha_{j,k}\}$ can be calculated as

$$\alpha_{j,k} = \int_{\mathbb{R}} f(x)\phi_{j,k}(x)dx \tag{3.23}$$

In other words, if the scaling functions are orthonormal, the terms in the single-scale approximation of f are simply the projections of f onto the various scaling functions $\{\phi_{j,k}\}$. In many cases, the integrals in equation (3.23) are not convenient to evaluate. Indeed, for some wavelet families, the functions are fractal in nature and have no known analytical form. Therefore, it is common in practice to assume that the single-scale coefficients are simply scaled samples of the function or signal f:

$$\alpha_{j,k} \sim f(2^{-j}k) \tag{3.24}$$

Equation (3.24) gives a reasonable approximation of the coefficients if j is large because the scaling functions tend to behave as Dirac delta functions when they are supported on fine-resolution grids.

Equation (3.17) implies that there is an equivalent, multiscale representation of f_j given by

$$f_j(x) = \sum_k \alpha_{j_0,k}\phi_{j_0,k}(x) + \sum_{l=j_0}^{j-1}\sum_k \beta_{l,k}\psi_{l,k}(x) \tag{3.25}$$

This multilevel expansion is in terms of wavelets on levels j_0 through $j-1$ and scaling functions on level j_0 only. The scaling function coefficients on the coarsest level represent a coarse-scale averaging of the function f. Meanwhile, the wavelet coefficients on the different levels represent details of varying resolution. These multiscale wavelet coefficients are known as the discrete wavelet transform of the function f_j. Note that, due to the compact support of the wavelets, the multilevel representation gives information not only about the frequency content of the function, but also spatial information. In other words, the wavelet coefficients are localized in space and give information about where certain features appear in the function. This property of spatial localization renders wavelets especially effective (and efficient) for analyzing functions or signals that exhibit transient behavior or have sharp features or discontinuities. More will be said about this in the next section in comparing wavelets to Fourier analysis. The multiscale expansion in equation (3.25) is often a very efficient representation in that many of the coefficients are close to or equal to zero. Often times, a function can be accurately represented using relatively few wavelet coefficients. Furthermore, the discrete wavelet transform can be calculated in a fast, efficient manner from the single-scale coefficients in equation (3.20). Later, this will be shown to be a fast-filter operation and the relevant decomposition formulas will be derived for various wavelet families.

To this point, there has not been much discussion as to how wavelets are designed. Indeed, there are a large number of wavelet families which are designed to exhibit various properties such as smoothness, approximation order, and orthogonality. Hence, there are a wide variety of approaches to wavelet design. One feature common to all wavelet families is that the scaling function and wavelet pair (ϕ, ψ) must satisfy the two-scale equations. This can be demonstrated by considering the following decomposition of the space V_1:

$$V_1 = V_0 \oplus W_0 \tag{3.26}$$

where the spaces are defined as

$$V_1 := span\left\{\phi_{1,k}\right\}_{k\in\mathbb{Z}} = span\left\{\sqrt{2}\phi\left(2(\cdot)-k\right)\right\}_{k\in\mathbb{Z}} \tag{3.27}$$

$$V_0 := span\left\{\phi(\cdot-k)\right\}_{k\in\mathbb{Z}} \tag{3.28}$$

$$W_0 := span\left\{\psi(\cdot-k)\right\}_{k\in\mathbb{Z}} \tag{3.29}$$

From equation (3.26), it is clear that $V_0 \subset V_1$ and $W_0 \subset V_1$. Therefore, the functions that form a basis for V_0 can be written as a linear combination of the functions that span V_1. The same is true for the wavelets that span W_0. Hence, a scaling function and wavelet pair (ϕ, ψ) must satisfy the following two-scale equations:

$$\phi(x) = \sum_s a_s \phi_{1,s}(x) = \sqrt{2}\sum_s a_s \phi(2x-s) \tag{3.30}$$

$$\psi(x) = \sum_s b_s \phi_{1,s}(x) = \sqrt{2}\sum_s b_s \phi(2x-s) \tag{3.31}$$

Note that equation (3.8) has been used to expand the functions $\{\phi_{1,s}\}$. The constants $\{a_s\}$ and $\{b_s\}$ are known as the scaling function and wavelet filters, respectively. Every scaling function and wavelet pair has a unique set of filters which essentially define the functions. The number and range of filter coefficients is dependent on the support of the particular scaling function or wavelet, which varies among wavelet families.

All wavelet families, by design, must satisfy the two-scale relationships. Hence, the design of a scaling function and wavelet pair entails finding solutions to equations (3.30) and (3.31). In addition, it is usually desired that the resulting functions satisfy additional conditions such as smoothness, approximation order, and orthogonality. These properties, as well as the two-scale equations, impose constraints on the filter coefficients. Therefore, one design approach is to find an appropriate set of filters that satisfy the relevant constraints. The filter coefficients can be viewed as degrees of freedom in the design process. As the number of filter coefficients that characterize a particular wavelet family is increased, the resulting functions can be designed to satisfy an increasing number of properties. However, increasing the number of coefficients also increases the support of the functions, which is usually not desirable. Therefore, the goal is to find a minimal set of filters that meet the design constraints. Once these filters are found, the resulting scaling function and wavelet pair is derived from the two-scale equations in a limiting procedure. This procedure entails first using the two-scale equations to calculate the function values at integer knots. Then, the function values are found at half-integer knots, followed by quarter-integer knots, and so forth. In the limit, the entire scaling function and wavelet can be derived. Daubechies used this procedure to design the first set of orthonormal, compactly-supported wavelets [70]. The drawback of this approach is that the scaling function and wavelet are not obtained in an analytical form. Instead, they are fractal functions because they are defined by a limiting procedure.

An alternative approach to wavelet design is to work directly with analytical functions. These functions are used to generate scaling functions and wavelets that satisfy the two-scale equations as well as any additional properties that are desired. In general, this is not a trivial task in the single-generator case. However, it is not difficult to design multiwavelet families using analytical functions. Multiwavelets are comprised of a set of wavelets $\{\psi^1, \ldots, \psi^r\}$ that are generated from a set of r scaling functions $\{\phi^1, \ldots, \phi^r\}$. In recent years, Donovan, Geronimo, and Hardin have shown that compactly-supported, orthonormal, piecewise-polynomial multiwavelets can be generated from the functions that span finite shift-invariant (FSI) spaces using the technique of intertwining multiresolution analyses [2,3]. This design approach is used in Chapter 4 to construct multiwavelets of arbitrary approximation order from the functions that span the classical finite element spaces. As a final comment, it should be noted that a great deal of wavelet design is done in the frequency domain. The conditions that the filter coefficients must satisfy can be expressed in the frequency domain using the Fourier transform. The filters can then be constructed in the frequency domain using digital filter design techniques.

3.2 Wavelets and the Approximation of Functions

In this section, wavelets are compared to two engineering tools that are commonly used for the approximation of functions, Fourier methods and the finite element method. The basic properties of Fourier series and finite element functions will be reviewed. It will be shown that, in some respects, wavelets represent an intermediate ground between these two approximation techniques. It will be demonstrated that wavelets are particularly effective in the analysis of some types of functions or signals that are not well represented by Fourier series or finite element functions. Therefore, wavelets provide an alternative tool for the approximation of a large class of functions and signals that are encountered in engineering practice.

3.2.1 Fourier Analysis

Fourier methods comprise one of the most common tools for the approximation of functions in engineering practice. They are used extensively in signal processing applications, particularly in the analysis of periodic functions or signals. The continuous Fourier transform of an analog function or signal f is given by

$$F(j\omega) = \int_{-\infty}^{\infty} f(x)e^{-j\omega x}dx \tag{3.32}$$

The Fourier transform yields a continuous, complex-valued function $F(j\omega)$ in the frequency domain. The original function can be recovered using the inverse Fourier transform:

$$f(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(j\omega)e^{j\omega x}d\omega \tag{3.33}$$

The complex exponential functions can be expanded using Euler's identity:

$$e^{j\omega x} = \cos(\omega x) + j\sin(\omega x) \tag{3.34}$$

Then, it is clear that the Fourier transform expresses a function f in terms of its projections onto sinusoids of continuously-varying frequency. In this manner, the Fourier transform gives the frequency spectrum of a function or signal.

The Fourier transform in equation (3.32) gives information about the frequency content of a function or signal in terms of a continuous frequency spectrum. It is common, in practice, to evaluate the frequency content of signals in terms of discrete frequencies instead. This gives rise to the complex Fourier series, which is used in the approximation of periodic functions. The complex Fourier series expansion of a function $f \in L^2_{per}[0, L]$, where $L^2_{per}[0, L]$ is the space of periodic, square-integrable functions with period L, takes the form

$$f(x) = \sum_{n=-\infty}^{\infty} c_n e^{jn\omega_0 x} \tag{3.35}$$

where $\{c_n\}$ are complex Fourier coefficients and ω_0 represents the fundamental frequency, defined as $\omega_0 := \frac{2\pi}{L}$ rad/s. The complex Fourier series approximates a periodic function in terms of its projections onto complex exponentials at discrete frequencies corresponding to integer multiples of the fundamental frequency ω_0. It is not difficult to show that the complex Fourier series basis functions $\{e^{jn\omega_0 x}\}_{n\in\mathbb{Z}}$ satisfy the following orthogonality condition:

$$\left\langle e^{jn\omega_0 x}, e^{jk\omega_0 x} \right\rangle_{L^2_{per}[0,L]} = \int_0^L e^{jn\omega_0 x} e^{-jk\omega_0 x} dx = L\delta_{k,n} \tag{3.36}$$

Equation (3.36), for the case where $k \neq n$, can be verified as follows:

$$\begin{aligned} \int_0^L e^{jn\omega_0 x} e^{-jk\omega_0 x} dx &= \int_0^L e^{j(n-k)\omega_0 x} dx \\ &= \frac{1}{j(n-k)\omega_0} e^{j(n-k)\omega_0 x}\Big|_0^L \\ &= \frac{1}{j(n-k)\omega_0} \left\{ e^{j(n-k)\omega_0 L} - 1 \right\} \end{aligned}$$

Using the fact that $\omega_0 = \frac{2\pi}{L}$, we have

$$\left\langle e^{jn\omega_0 x}, e^{jk\omega_0 x} \right\rangle_{L^2_{per}[0,L]} = \frac{L}{j(n-k)2\pi} \left\{ e^{j(n-k)2\pi} - 1 \right\}$$

Using Euler's identity, we can write

$$e^{j(n-k)2\pi} = \cos\left[(n-k)2\pi\right] + j\sin\left[(n-k)2\pi\right] = 1$$

Then, we obtain the result

$$\left\langle e^{jn\omega_0 x}, e^{jk\omega_0 x} \right\rangle_{L^2_{per}[0,L]} = 0 \quad \text{for } k \neq n \tag{3.37}$$

If $k = n$, we have

$$\left\langle e^{jn\omega_0 x}, e^{jn\omega_0 x} \right\rangle_{L^2_{per}[0,L]} = \int_0^L e^{jn\omega_0 x} e^{-jn\omega_0 x} dx = \int_0^L dx = L \tag{3.38}$$

Hence, the functions $\{e^{jn\omega_0 x}\}$ form an orthogonal basis for the space $L^2_{per}[0, L]$.

It is not difficult to derive an expression for the complex Fourier coefficients $\{c_n\}$. This can be accomplished by taking the $L^2_{per}[0, L]$ inner product of both sides of equation (3.35) with the function $e^{jk\omega_0 x}$:

$$\int_0^L f(x) e^{-jk\omega_0 x} dx = \sum_{n \in \mathbb{Z}} c_n \int_0^L e^{jn\omega_0 x} e^{-jk\omega_0 x} dx$$

Using equation (3.36), this can be reduced to

$$\int_0^L f(x) e^{-jk\omega_0 x} dx = \sum_{n \in \mathbb{Z}} c_n L \delta_{k,n}$$

Finally, we obtain

$$c_n = \frac{1}{L} \int_0^L f(x) e^{-jn\omega_0 x} dx, \quad n \in \mathbb{Z} \tag{3.39}$$

Note that the complex Fourier series extends over both the positive and negative integers ($n \in \mathbb{Z}$). The coefficients $\{c_n\}$ and $\{c_{-n}\}$ form complex conjugate pairs since, using Euler's identity:

$$c_n = \frac{1}{L} \int_0^L f(x) e^{-jn\omega_0 x} dx = \frac{1}{L} \int_0^L f(x) \cos(n\omega_0 x) dx - j \frac{1}{L} \int_0^L f(x) \sin(n\omega_0 x) dx$$

$$c_{-n} = \frac{1}{L} \int_0^L f(x) e^{jn\omega_0 x} dx = \frac{1}{L} \int_0^L f(x) \cos(n\omega_0 x) dx + j \frac{1}{L} \int_0^L f(x) \sin(n\omega_0 x) dx$$

Clearly, then, $c_{-n} = \overline{c_n}$.

Using Euler's identity once again, the complex Fourier series

$$f(x) = \sum_{n\in\mathbb{Z}} c_n e^{jn\omega_0 x} \tag{3.40}$$

can be expanded into the form

$$\begin{aligned} f(x) &= \sum_{n\in\mathbb{Z}} c_n \left\{\cos(n\omega_0 x) + j\sin(n\omega_0 x)\right\} \\ &= c_0 + \sum_{n=1}^{\infty} c_n \left\{\cos(n\omega_0 x) + j\sin(n\omega_0 x)\right\} \\ &\quad + \sum_{n=-1}^{-\infty} c_n \left\{\cos(n\omega_0 x) + j\sin(n\omega_0 x)\right\} \end{aligned} \tag{3.41}$$

Using the fact that the coefficients form complex conjugate pairs, the sum over the negative integers can be rewritten as

$$\begin{aligned} \sum_{n=-1}^{-\infty} c_n \left\{\cos(n\omega_0 x) + j\sin(n\omega_0 x)\right\} &= \sum_{n=1}^{\infty} c_{-n} \left\{\cos(n\omega_0 x) - j\sin(n\omega_0 x)\right\} \\ &= \sum_{n=1}^{\infty} \overline{c_n} \left\{\cos(n\omega_0 x) - j\sin(n\omega_0 x)\right\} \end{aligned}$$

Then, equation (3.41) takes the form

$$f(x) = c_0 + \sum_{n=1}^{\infty} \left\{c_n + \overline{c_n}\right\} \cos(n\omega_0 x) + \sum_{n=1}^{\infty} \left\{c_n - \overline{c_n}\right\} j\sin(n\omega_0 x) \tag{3.42}$$

Defining

$$\begin{aligned} a_0 &= c_0 \\ a_n &= c_n + \overline{c_n} = 2\mathrm{Re}(c_n) \\ b_n &= c_n - \overline{c_n} = 2\mathrm{Im}(c_n) \end{aligned} \tag{3.43}$$

where $\mathrm{Re}(c_n)$ and $\mathrm{Im}(c_n)$ denote the real and imaginary parts of the coefficient c_n, we obtain the real Fourier series:

$$f(x) = a_0 + \sum_{n=1}^{\infty} a_n \cos(n\omega_0 x) + \sum_{n=1}^{\infty} b_n \sin(n\omega_0 x) \tag{3.44}$$

Note that the coefficients $\{a_n\}$ and $\{b_n\}$ are all real-valued. The real form of the Fourier series is a commonly used alternative to the complex Fourier series. From equations (3.42) and (3.43), the real Fourier coefficients are given by

$$a_0 = \frac{1}{L}\int_0^L f(x)dx \tag{3.45}$$

$$a_n = \frac{2}{L}\int_0^L f(x)\cos(n\omega_0 x)dx, \quad n \in \mathbb{Z}^+ \tag{3.46}$$

$$b_n = \frac{2}{L}\int_0^L f(x)\sin(n\omega_0 x)dx, \quad n \in \mathbb{Z}^+ \tag{3.47}$$

where $\mathbb{Z}^+$ denotes the positive integers. Note that the Fourier basis functions satisfy the following orthogonality properties:

$$\begin{aligned}
&(1) \qquad \int_0^{2L} \cos(m\omega_0 x)\cos(n\omega_0 x)dx = \delta_{m,n} \\
&(2) \qquad \int_0^{2L} \sin(m\omega_0 x)\sin(n\omega_0 x)dx = \delta_{m,n} \\
&(3) \qquad \int_0^{2L} \cos(m\omega_0 x)\sin(n\omega_0 x)dx = 0 \quad \forall\ m, n \in \mathbb{Z}^+
\end{aligned} \tag{3.48}$$

Therefore, the Fourier basis functions form an orthogonal basis for $L^2_{per}(\mathbb{R})$.

Fourier methods provide useful information about the frequency content of functions or signals. One drawback, however, is that since the Fourier basis functions have global support (they are supported over the entire real line), they are not capable of providing spatial information about a function. That is, the Fourier transform gives the spectral content of a signal, but cannot give information about where features

occur in the signal in the spatial (or time) domain. Therefore, Fourier methods are not always optimal for the analysis of time-varying functions whose spectral content changes with time. In addition, Fourier methods are not particularly efficient in representing signals with sharp features or discontinuities. This is because the basis functions have global support. So, in representing a square pulse, as an example, it takes many terms in the Fourier series to average the contributions of the basis functions to zero outside of the square. A distinguishing characteristic of a Fourier series approximation of a signal with sharp edges is an overshoot where these features occur. Therefore, Fourier methods provide a valuable tool for analyzing the frequency content of many signals, but also have limitations for a wide class of signals that occur in engineering practice.

3.2.2 Finite Element Analysis

In contrast to the Fourier transform, which gives the spectral content of a function or signal, finite element methods (FEM) provide details about a function in the spatial domain. In the finite element method, the spatial domain is discretized into a number of elements, and the nodes of these elements represent degrees of freedom in the system. Therefore, a continuous function, such as the displacement of a beam in bending, is approximated in terms of a number of discrete degrees of freedom. In using this technique, the governing equation of the system, typically a partial differential equation, is discretized into a matrix equation which is then solved for the nodal degrees of freedom. Clearly, the accuracy of such a solution depends on the number of elements used. More elements allow for higher accuracy, but also require more computational effort to generate a solution. The key to the finite element method is that the basis functions have local support, meaning that they are supported over only a few elements. Thus, the resulting matrix equation is relatively sparse and block diagonal in structure. Another important feature of finite element basis functions is that they are interpolatory at the nodes. This means that at each node one

basis function has a value of one while the others have values of zero. Hence, in the resulting approximation of the function, the coefficients are actually the nodal values.

A finite element approximation of a one-dimensional function f takes the form

$$f_h(x) = \sum_i d_i N_{h,i}(x) \tag{3.49}$$

where h denotes the discretization level. Small values of h denote finer meshes while larger values of h correspond to coarser meshes. The $\{d_i\}$ are the expansion coefficients, which usually represent physical values (such as nodal displacements) since the basis functions are interpolatory at the nodes. The basis functions $\{N_{h,i}\}$ are typically piecewise-polynomial in form. Classical finite element basis functions are derived from Lagrange polynomials, which interpolate a series of equally-spaced nodes on each element [100]. The number of nodes in an element is equivalent to the number of basis functions over that element. For example, classical linear basis functions are obtained by applying Lagrange interpolation to a two-node element. The resulting elemental basis functions $\{N_1, N_2\}$ are given by

$$N_1(x) = 1 - \frac{x}{L} \tag{3.50}$$

$$N_2(x) = \frac{x}{L} \tag{3.51}$$

where L is the elemental length. These functions are shown in Figure (3.2). When several elements are assembled, we obtain the classical hat functions which are used for modeling the axial displacement of a rod, as an example. These linear functions have approximation order of one, meaning that they can exactly represent all first-order (linear) polynomials. The classical finite element functions can be constructed to have any desired approximation order by varying the number of nodes. For example, applying the Lagrange interpolation over a three-node element yields three quadratic basis functions of approximation order two. It should also be noted that the finite

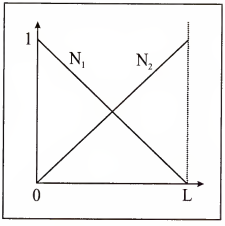


Figure 3.2: Linear Finite Element Basis Functions

element basis functions are not typically orthogonal. As an example, one can easily verify that the functions N_1 and N_2 in Figure (3.2) are not orthogonal over the element.

As stated before, the accuracy of the finite element approximation in equation (3.49) depends on the mesh parameter h. The mesh parameter can be thought of as defining a frequency for the basis functions, similar to the dilation index j for wavelets. Finite element bases generate approximations via nested spaces. A fine-scale space V_h can be defined as

$$V_h := span\left\{N_{h,k}\right\}_{k\in\mathbb{Z}} \tag{3.52}$$

The space V_{2h} is spanned by basis functions that are supported over a mesh that is twice as coarse as h. It is not difficult to show that

$$V_{2h} \subset V_h \tag{3.53}$$

In general, the nested spaces take the form

$$V_{20h} \subset V_{19h} \subset \cdots \subset V_h \tag{3.54}$$

In the limit, the spaces $\{V_h\}$ approach $\{0\}$ as the mesh size approaches infinity and $L^2(\mathbb{R})$ as the mesh size approaches zero. The nested character of the finite element spaces is similar in form to that for the spaces $\{V_j\}$ that are encountered in a wavelet-based multiresolution analysis. The main difference is that the finite element basis functions are not amenable to multiresolution analysis. This is due in large part to the fact that, unlike wavelets and their associated scaling functions, there are no convenient relationships between the functions that span the different spaces $\{V_h\}$. It should be noted that multigrid methods are commonly used to generate finite element solutions by iterating over different mesh sizes. This is not the same as multiresolution analysis, however, which yields a multiscale representation in terms of localized functions (wavelets) of differing frequencies. Therefore, in general, finite element methods are not particularly useful for obtaining information about a function in the frequency domain.

3.2.3 Wavelets

In previous sections, it was demonstrated that Fourier methods provide valuable information about the spectral content of a function or signal, but are unable to give spatial information about where features occur. In contrast, finite element methods can yield spatial information to arbitrary accuracy, but do not provide frequency information. Wavelets are, in a sense, intermediate to these two common engineering tools. It has been shown that wavelets and their associated multiresolution analyses provide information about the frequency content of functions or signals. The multilevel wavelet basis is similar to the Fourier basis in that it is comprised of functions of many different frequencies. However, while Fourier basis functions have global support, wavelets are compactly supported. In this respect, they are similar to finite element functions and can provide information in the spatial domain. The compact support of the wavelets enables efficient representations of signals with sharp edges or discontinuities. There is no need to average out the contributions of functions away

from the features as is the case with Fourier series. For example, in approximating a square pulse, all of the wavelet coefficients away from the pulse are zero, and only a few localized functions contribute to the representation. Therefore, wavelets combine some of the advantages of Fourier analysis and finite element methods.

Similar to the continuous Fourier transform, there is a continuous wavelet transform, defined as

$$F(a,b) = \int_{\mathbb{R}} f(x)\psi\left(\frac{x-a}{b}\right) dx \tag{3.55}$$

The inverse wavelet transform is given by

$$f(x) = \int_{\mathbb{R}}\int_{\mathbb{R}} F(a,b)\psi\left(\frac{x-a}{b}\right) da\, db \tag{3.56}$$

The continuous wavelet transform projects a function onto a continuous series of translates and dilates of a wavelet ψ. The index $a \in \mathbb{R}$ represents a continuous translation index, while the index $b \in \mathbb{R}$ denotes a continuous frequency parameter. In practice, the wavelet transform is calculated over a series of integer translates k and discrete frequencies 2^j, $j \in \mathbb{Z}$. This leads to the discrete wavelet transform, which gives a multiscale representation of a function or signal. As discussed in Section 3.1, this multilevel expansion takes the form

$$f_j(x) = \sum_k \alpha_{j_0,k}\phi_{j_0,k}(x) + \sum_{l=j_0}^{j-1}\sum_k \beta_{l,k}\psi_{l,k}(x) \tag{3.57}$$

where f_j denotes a fine-scale approximation of the original function f on resolution level j. If the wavelets are orthonormal, the expansion coefficients can be calculated as the projections of the function onto each wavelet:

$$\beta_{j,k} = \int_{\mathbb{R}} f(x)\psi_{j,k}(x)dx \tag{3.58}$$

It will be shown later that, in practice, the coefficients $\{\beta_{j,k}\}$ are not calculated in this manner but rather from a recursive application of fast decomposition formulas. Note the similarity in form between the Fourier series and the multilevel wavelet expansion given in equation (3.57). If the wavelets are orthonormal, the expansion coefficients can be calculated in the same manner as the Fourier coefficients.

It has already been noted that the nested spaces $\{V_j\}$ that form a multiresolution analysis are similar to the nested finite element spaces $\{V_h\}$. A fundamental difference is that, with wavelets, the two-scale relationships provide a convenient means of relating the functions on different levels. This property renders wavelets amenable to multiresolution analysis, while finite element functions are not. In comparing wavelets and finite element functions, it is interesting to note that wavelets also can be constructed to have arbitrary approximation order. While for classical finite element functions, approximation order is determined by the number of interpolated nodes, for wavelets it is related to the concept of vanishing moments. The moments of a wavelet are defined as

$$M_p := \int_{\mathbb{R}} x^p \psi(x) dx \tag{3.59}$$

The number of vanishing moments is the largest value of p for which the moment M_p is zero. It can be demonstrated that the approximation order of a wavelet is equivalent to its number of vanishing moments [99]. If a wavelet has N vanishing moments, then it has approximation order N and the wavelets can exactly reproduce polynomials of order N. Note that the zero moment M_0 vanishes for all wavelets, meaning that wavelets have an average value of zero over the real line.

In conclusion, wavelets possess some of the desirable properties of Fourier series and finite element methods. An effort has been made in this section to clarify the similarities and differences between these three engineering tools. It is important to note that, while wavelets provide an alternative analysis tool for functions and signals

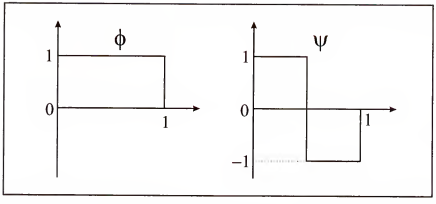


Figure 3.3: Haar Scaling Function and Wavelet

for which Fourier or finite element methods are not optimal, there are many cases where wavelets are not necessarily the best choice. It is clear, however, that wavelets fill a unique niche in the spectrum of approximation tools in engineering practice.

3.3 The Haar Wavelet

Section 3.1 has provided a brief introduction to wavelets and multiresolution analysis. This section elaborates upon some of these ideas, focusing on the Haar wavelet family. While the Haar wavelet is seldom used in practice because it is a discontinuous function with low approximation order, it is worth studying because the concepts presented in Section 3.1 are easily visualized for this simple function. This is not usually the case for more complex wavelet families. Therefore, the Haar wavelet serves as an excellent prototype for studying wavelets. The discussion can then be extended to orthonormal wavelet families and, more generally, orthonormal multiwavelets.

Figure (3.3) shows the Haar wavelet and scaling function defined on the interval $[0, 1]$. The scaling function ϕ shown in Figure (3.3) is the generator from which the wavelet ψ is constructed. The Haar scaling function and wavelet are defined as

$$\phi(x) := \begin{cases} 1 & 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.60}$$

$$\psi(x) := \begin{cases} 1 & 0 \leq x < \frac{1}{2} \\ -1 & \frac{1}{2} \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.61}$$

The level on which the Haar scaling function and wavelet are defined is denoted as level 0. Recall that the scaling functions and wavelets on different resolution levels are related to the functions defined in equations (3.60) and (3.61) by the expressions

$$\phi_{j,k}(x) = 2^{j/2}\phi\left(2^j x - k\right) \tag{3.62}$$

$$\psi_{j,k}(x) = 2^{j/2}\psi\left(2^j x - k\right) \tag{3.63}$$

where $j \in \mathbb{Z}$ is the dilation index, $k \in \mathbb{Z}$ is the translation index, and the $2^{j/2}$ term is a normalization factor. Thus, the functions in equations (3.62) and (3.63) are scaled dilates and translates of the original scaling function and wavelet. Note that, evaluating the functions $\phi_{j,k}$ and $\psi_{j,k}$ using the definitions of ϕ and ψ, we have

$$\phi_{j,k}(x) = \begin{cases} 2^{j/2} & 2^{-j}k \leq x \leq 2^{-j}(k+1) \\ 0 & \text{otherwise} \end{cases} \tag{3.64}$$

$$\psi_{j,k}(x) = \begin{cases} 2^{j/2} & 2^{-j}k \leq x < 2^{-j}(k+\frac{1}{2}) \\ -2^{j/2} & 2^{-j}(k+\frac{1}{2}) \leq x \leq 2^{-j}(k+1) \\ 0 & \text{otherwise} \end{cases} \tag{3.65}$$

Several scaling functions and wavelets on different resolution levels are plotted in Figure (3.4) over the interval $[0, 1]$. For clarity, only every other translate is shown.

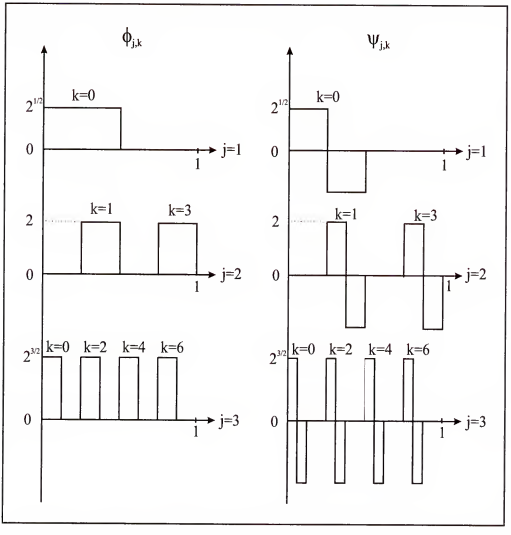


Figure 3.4: Scaling Functions and Wavelets on Different Resolution Levels

Note that the functions on the different levels have different amplitudes due to the $2^{j/2}$ normalization factor.

The functions $\{\phi_{j,k}\}$ and $\{\psi_{j,k}\}$ satisfy the following orthogonality relationships:

$$
\begin{aligned}
&(1) \qquad \langle \phi_{j,k}, \phi_{j,m} \rangle = \int_{\mathbb{R}} \phi_{j,k}(x)\phi_{j,m}(x)dx = \delta_{k,m} \\
&(2) \qquad \langle \psi_{j,k}, \psi_{l,m} \rangle = \int_{\mathbb{R}} \psi_{j,k}(x)\psi_{l,m}(x)dx = \delta_{(j,k),(l,m)} \\
&(3) \qquad \langle \phi_{j,k}, \psi_{l,m} \rangle = \int_{\mathbb{R}} \phi_{j,k}(x)\psi_{l,m}(x)dx = 0 \qquad \forall\ k, m \in \mathbb{Z},\ j \leq l
\end{aligned}
\tag{3.66}
$$

The notation $\delta_{(j,k),(l,m)}$ is defined as

$$\delta_{(j,k),(l,m)} := \delta_{j,l}\delta_{k,m} = \begin{cases} 1 & j = l \text{ and } k = m \\ 0 & \text{otherwise} \end{cases} \tag{3.67}$$

The first orthogonality property in equation (3.66) states that the scaling functions on the same level are orthonormal. By inspection, it is clear that these functions are orthogonal since no two scaling functions on the same level have overlapping support. The orthonormality of the scaling functions can be verified by taking the $L^2(\mathbb{R})$ norm of an arbitrary scaling function $\phi_{j,k}$:

$$\|\phi_{j,k}\|_{L^2(\mathbb{R})} = \left[\langle \phi_{j,k}, \phi_{j,k} \rangle_{L^2(\mathbb{R})}\right]^{\frac{1}{2}} \tag{3.68}$$

Making use of equation (3.64), the inner product $\langle \phi_{j,k}, \phi_{j,k} \rangle$ is given by

$$\begin{aligned} \langle \phi_{j,k}, \phi_{j,k} \rangle &= \int_{\mathbb{R}} \phi_{j,k}(x)\phi_{j,k}(x)dx \\ &= \int_{2^{-j}k}^{2^{-j}(k+1)} 2^j dx = 2^j x\Big|_{2^{-j}k}^{2^{-j}(k+1)} \\ &= 2^j\left[2^{-j}(k+1) - 2^{-j}k\right] \\ &= (k+1) - k \\ &= 1 \end{aligned}$$

Therefore, it is clear that

$$\|\phi_{j,k}\|_{L^2(\mathbb{R})} = 1$$

and the scaling functions $\{\phi_{j,k}\}_{k\in\mathbb{Z}}$ on a given level j form an orthonormal set. The second orthogonality property states that two given wavelets are orthogonal not only if they are supported on the same level, but also if they are on different levels. This

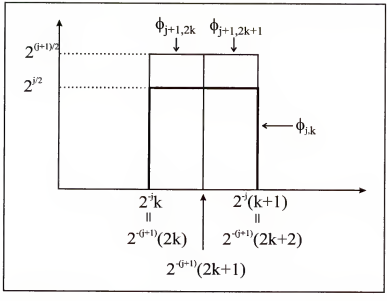


Figure 3.5: Haar Scaling Functions on Adjacent Resolution Levels

can be seen by inspection, and the orthonormality of the wavelets can be demonstrated in a similar manner to that used for the scaling functions. Finally, the third orthogonality property states that the scaling functions are orthogonal to the wavelets that are supported on the same level. In addition, a given scaling function is orthogonal to any wavelet that is supported on a finer grid. This property can easily be verified by inspection.

Recall from Section 3.1 that there is a relationship between the scaling functions that are supported on one level to those supported on an adjacent resolution level. Figure (3.5) shows the kth translate of a scaling function $\phi_{j,k}$ on an arbitrary level j. Also shown are two scaling functions, $\phi_{j+1,2k}$ and $\phi_{j+1,2k+1}$, supported on level $j+1$, a grid that is twice as fine as level j. The support of the functions $\phi_{j+1,2k}$ and $\phi_{j+1,2k+1}$ overlap that of $\phi_{j,k}$. The translation indices of these functions are $2k$ and $2k+1$ due to the fact that the grid is now twice as fine. Note that the scaling functions on level j have an amplitude of $2^{j/2}$ while those on level $j+1$ have an amplitude of $2^{(j+1)/2}$. Therefore, the act of contracting a scaling function (or wavelet) results in an increase in the amplitude of that function. From Figure (3.5), it can be seen that the scaling

function $\phi_{j,k}$ can be expressed as a weighted sum of the scaling functions $\phi_{j+1,2k}$ and $\phi_{j+1,2k+1}$ as follows:

$$\phi_{j,k}(x) = \tfrac{1}{\sqrt{2}}\phi_{j+1,2k}(x) + \tfrac{1}{\sqrt{2}}\phi_{j+1,2k+1}(x) \tag{3.69}$$

The $\frac{1}{\sqrt{2}}$ factors in equation (3.69) take into account the difference in amplitude between the scaling functions on level j with those supported on level $j+1$.

In a similar manner, wavelets on level j can be constructed from scaling functions that are supported on level $j+1$. It is not difficult to see that the wavelet $\psi_{j,k}$ can be formed (modulo a constant) by piecing together a positive and a negative scaling function on level $j+1$. The function $\psi_{j,k}$ can be expressed as a linear combination of the scaling functions $\phi_{j+1,2k}$ and $\phi_{j+1,2k+1}$ as follows:

$$\psi_{j,k}(x) = \tfrac{1}{\sqrt{2}}\phi_{j+1,2k}(x) - \tfrac{1}{\sqrt{2}}\phi_{j+1,2k+1}(x) \tag{3.70}$$

Together, equations (3.69) and (3.70) comprise the generalized two-scale equations for the Haar scaling function and wavelet. These equations show that the scaling functions and wavelets on level j can be constructed as weighted combinations of the scaling functions that are supported on the finer-resolution level $j+1$. As an example, setting $j=0$ and $k=0$ in equations (3.69) and (3.70), we obtain

$$\begin{aligned} \phi(x) &= \tfrac{1}{\sqrt{2}}\phi_{1,0}(x) + \tfrac{1}{\sqrt{2}}\phi_{1,1}(x) \\ &= \phi(2x) + \phi(2x-1) \qquad (3.71) \\ \psi(x) &= \tfrac{1}{\sqrt{2}}\phi_{1,0}(x) - \tfrac{1}{\sqrt{2}}\phi_{1,1}(x) \\ &= \phi(2x) - \phi(2x-1) \qquad (3.72) \end{aligned}$$

The Haar scaling function ϕ is the generator of a series of nested spaces $\{V_j\}_{j\in\mathbb{Z}}$ that form a multiresolution analysis for $L^2(\mathbb{R})$. Each space V_j is defined as

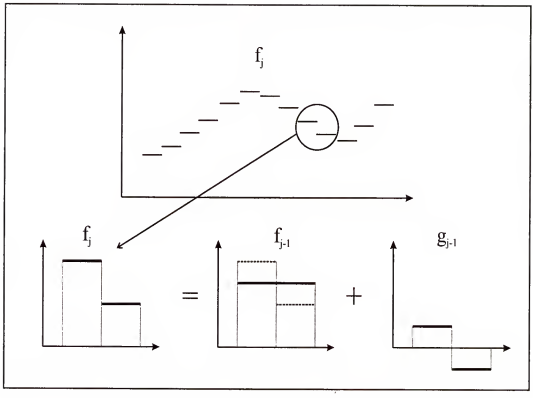


Figure 3.6: Decomposition of V_j into V_{j-1} and W_{j-1}

$$V_j := span\left\{\phi_{j,k}\right\}_{k\in\mathbb{Z}} \tag{3.73}$$

Recall that the space V_j can be decomposed as

$$V_j = V_{j-1} \oplus W_{j-1} \tag{3.74}$$

where

$$V_{j-1} := span\left\{\phi_{j-1,k}\right\}_{k\in\mathbb{Z}} \tag{3.75}$$

and W_{j-1} is the orthogonal complement space, defined as

$$W_{j-1} := span\left\{\psi_{j-1,k}\right\}_{k\in\mathbb{Z}} \tag{3.76}$$

The wavelet space W_{j-1} represents the difference between the two adjacent spaces V_j and V_{j-1}. To visualize this, consider the function f, shown in Figure (3.6), which has been approximated in terms of the scaling functions $\{\phi_{j,k}\}_{k\in\mathbb{Z}}$ that span the space V_j. This simply amounts to approximating f using piecewise constants on a grid with spacing 2^{-j}. Consider the circled region that has been enlarged in the figure. The function f_j represents the approximation of the function f in terms of V_j, which can be written as a sum of the functions $f_{j-1} \in V_{j-1}$ and $g_{j-1} \in W_{j-1}$. Note that f_{j-1} is the average of f_j while g_{j-1} represents the difference between f_j and f_{j-1}. Equation (3.74) can be applied recursively to obtain the following multiscale decomposition of the space V_j:

$$V_j = V_{j_0} \oplus W_{j_0} \oplus W_{j_0-1} \oplus \cdots \oplus W_{j-2} \oplus W_{j-1} \tag{3.77}$$

where j_0 is the coarsest level used in the approximation.

Recall that equation (3.73) gives a single-scale approximation of a function $f \in L^2(\mathbb{R})$. This representation takes the form

$$f_j(x) = \sum_{k\in\mathbb{Z}} \alpha_{j,k}\phi_{j,k}(x) \tag{3.78}$$

where $\{\alpha_{j,k}\}$ are constant scaling function coefficients and j is usually chosen to correspond to a relatively fine grid. For the Haar scaling function, f_j is simply a piecewise-constant approximation of f. Equation (3.74) states that an equivalent two-scale expansion of f_j can be written as

$$f_j(x) = \sum_{k\in\mathbb{Z}} \alpha_{j-1,k}\phi_{j-1,k}(x) + \sum_{k\in\mathbb{Z}} \beta_{j-1,k}\psi_{j-1,k}(x) \tag{3.79}$$

where $\{\alpha_{j-1,k}\}$ and $\{\beta_{j-1,k}\}$ represent constant scaling function and wavelet coefficients. Finally, equation (3.77) gives a multiscale representation of f_j of the form

$$f_j(x) = \sum_{k\in\mathbb{Z}} \alpha_{j_0,k}\phi_{j_0,k}(x) + \sum_{l=j_0}^{j-1}\sum_{k\in\mathbb{Z}} \beta_{l,k}\psi_{l,k}(x) \tag{3.80}$$

This multilevel expansion is in terms of the scaling functions on the coarsest level j_0 and the wavelets that are supported on levels j_0 through $j-1$. Recall that this representation is desirable because it enables both spatial and frequency resolution of a function or signal. In addition, it is often the case that a large number of the multiscale coefficients are close to zero and can be neglected. This leads to a relatively sparse representation of the function.

As an example, the Haar wavelet transform can be used to decompose the function depicted in Figure(3.7). This function is a .5 Hz sine wave over the domain $[0,1]$ with a sharp feature in the form of a 32 Hz sine component. Figure (3.8) shows the Haar wavelet decomposition of this function. The original fine-scale representation of the function is taken on level 8, which corresponds to an approximation in terms of piecewise constants of width 2^{-8}. The coarsest level taken in the decomposition is level 0. Therefore, we have a multilevel wavelet representation of the function in terms of the scaling function approximation on level 0, which corresponds to the average value of the function over $[0,1]$, and the wavelet details on levels 0 through 7. Note that, on the finer-resolution levels, the wavelet details are nonzero only where the sharp features of the function occur. On the coarser scales, there are no details from these sharp features and the wavelet details focus on the low frequency component of the signal.

In practice, the single-scale approximation is the starting point for a multiresolution analysis. A fundamental issue, then, is how to calculate the coefficients that appear in the multiscale representation in equation (3.80). In this discussion, it is assumed that the single-scale coefficients $\{\alpha_{j,k}\}$ have been determined, perhaps using one of the techniques discussed in Section 3.1. It is not difficult to derive decomposition formulas from which the multiscale coefficients can be calculated from the

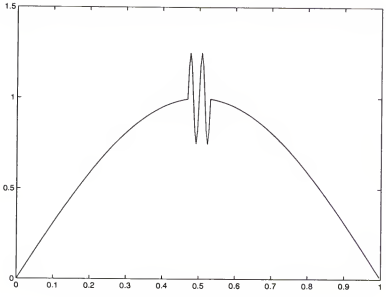


Figure 3.7: Original Function

single-scale coefficients $\{\alpha_{j,k}\}$. In order to do this, we first note that the single-scale and two-scale representations of a function f are equivalent. From equations (3.78) and (3.79), we have

$$\sum_{k\in\mathbb{Z}} \alpha_{j,k}\phi_{j,k}(x) = \sum_{k\in\mathbb{Z}} \alpha_{j-1,k}\phi_{j-1,k}(x) + \sum_{k\in\mathbb{Z}} \beta_{j-1,k}\psi_{j-1,k}(x) \tag{3.81}$$

Taking the $L^2(\mathbb{R})$ inner product of both sides of equation (3.81) with $\phi_{j-1,m}$, an arbitrary scaling function on level $j-1$, we obtain

$$\begin{aligned}\sum_{k\in\mathbb{Z}} \alpha_{j,k} \int_{\mathbb{R}} \phi_{j,k}(x)\phi_{j-1,m}(x)dx &= \sum_{k\in\mathbb{Z}} \alpha_{j-1,k} \int_{\mathbb{R}} \phi_{j-1,k}(x)\phi_{j-1,m}(x)dx \\ &\quad + \sum_{k\in\mathbb{Z}} \beta_{j-1,k} \int_{\mathbb{R}} \psi_{j-1,k}(x)\phi_{j-1,m}(x)dx\end{aligned} \tag{3.82}$$

Recall that, from the orthogonality properties of the Haar scaling function and wavelet, the integrals on the right-hand side of equation (3.82) can be simplified as

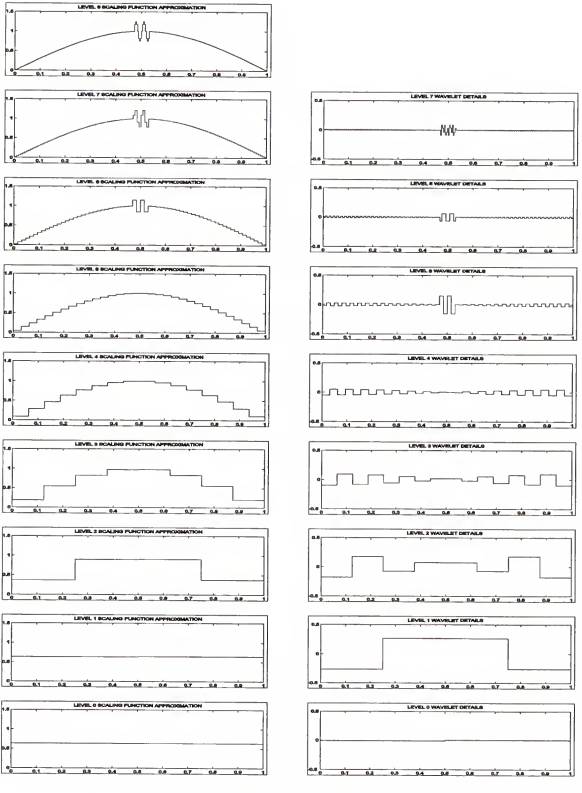


Figure 3.8: Haar Wavelet Decomposition

$$\begin{aligned}\int_{\mathbb{R}} \phi_{j-1,k}(x)\phi_{j-1,m}(x)dx &= \delta_{k,m} \\ \int_{\mathbb{R}} \psi_{j-1,k}(x)\phi_{j-1,m}(x)dx &= 0 \qquad \forall\ k,m \in \mathbb{Z}\end{aligned}$$

Then, equation (3.82) reduces to

$$\sum_k \alpha_{j-1,k}\delta_{k,m} = \alpha_{j-1,m} = \sum_{k\in\mathbb{Z}} \alpha_{j,k} \int_{\mathbb{R}} \phi_{j,k}(x)\phi_{j-1,m}(x)dx$$

Substituting for $\phi_{j-1,m}$ using the two-scale relationship, equation (3.69), we obtain

$$\alpha_{j-1,m} = \sum_{k\in\mathbb{Z}} \alpha_{j,k} \int_{\mathbb{R}} \phi_{j,k}(x) \left[\tfrac{1}{\sqrt{2}}\phi_{j,2m}(x) + \tfrac{1}{\sqrt{2}}\phi_{j,2m+1}(x)\right] dx$$

Using the orthogonality of the scaling functions once again, this can be written as

$$\alpha_{j-1,m} = \sum_k \alpha_{j,k} \tfrac{1}{\sqrt{2}} \left(\delta_{k,2m} + \delta_{k,2m+1}\right)$$

Finally, the following expression for $\alpha_{j-1,m}$ is obtained:

$$\alpha_{j-1,m} = \tfrac{1}{\sqrt{2}} \left(\alpha_{j,2m} + \alpha_{j,2m+1}\right) \tag{3.83}$$

In a similar manner, taking the inner product of both sides of equation (3.81) with $\psi_{j-1,m}$, we have

$$\begin{aligned}\sum_{k\in\mathbb{Z}} \alpha_{j,k} \int_{\mathbb{R}} \phi_{j,k}(x)\psi_{j-1,m}(x)dx &= \sum_{k\in\mathbb{Z}} \alpha_{j-1,k} \int_{\mathbb{R}} \phi_{j-1,k}(x)\psi_{j-1,m}(x)dx \\ &\quad + \sum_{k\in\mathbb{Z}} \beta_{j-1,k} \int_{\mathbb{R}} \psi_{j-1,k}(x)\psi_{j-1,m}(x)dx\end{aligned}$$

which, by orthogonality, reduces to

$$\beta_{j-1,m} = \sum_{k\in\mathbb{Z}} \alpha_{j,k} \int_{\mathbb{R}} \phi_{j,k}(x)\psi_{j-1,m}(x)dx$$

Then, substituting for $\psi_{j-1,m}$ using the two-scale equation, equation (3.70), we obtain

$$\beta_{j-1,m} = \sum_{k\in\mathbb{Z}} \alpha_{j,k} \int_{\mathbb{R}} \phi_{j,k}(x) \left[\tfrac{1}{\sqrt{2}}\phi_{j,2m}(x) - \tfrac{1}{\sqrt{2}}\phi_{j,2m+1}(x)\right] dx$$

Once again making use of the orthogonality of the scaling functions, this can be simplified to

$$\beta_{j-1,m} = \sum_{k} \alpha_{j,k} \tfrac{1}{\sqrt{2}} \left(\delta_{k,2m} - \delta_{k,2m+1}\right)$$

Finally, the following expression for $\beta_{j-1,m}$ is obtained:

$$\beta_{j-1,m} = \tfrac{1}{\sqrt{2}} \left(\alpha_{j,2m} - \alpha_{j,2m+1}\right) \tag{3.84}$$

Equations (3.83) and (3.84) are decomposition formulas whereby the coefficients $\{\alpha_{j-1,k}\}$ and $\{\beta_{j-1,k}\}$ can be calculated from the single-scale coefficients $\{\alpha_{j,k}\}$ in a fast, efficient manner. Clearly, the coefficients $\{\alpha_{j-2,k}\}$ and $\{\beta_{j-2,k}\}$ can then be determined from $\{\alpha_{j-1,k}\}$. Continuing in this manner, the entire array of multilevel coefficients (the discrete wavelet transform) can be calculated through a recursive application of the decomposition formulas. Hence, generating the coefficients in the multiscale representation of a function is a recursive, fast-filter operation implemented over all the levels that are included in the decomposition. It is convenient to express the results in equations (3.83) and (3.84) in matrix form:

$$\begin{Bmatrix} \vdots \\ \alpha_{j-1,m} \\ \alpha_{j-1,m+1} \\ \alpha_{j-1,m+2} \\ \vdots \end{Bmatrix} = \begin{bmatrix} \ddots & & & & & & & \\ & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & \\ & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 & 0 & \\ & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \\ & & & & & & & \ddots \end{bmatrix} \begin{Bmatrix} \vdots \\ \alpha_{j,2m} \\ \alpha_{j,2m+1} \\ \alpha_{j,2m+2} \\ \vdots \end{Bmatrix} \tag{3.85}$$

$$\begin{Bmatrix} \vdots \\ \beta_{j-1,m} \\ \beta_{j-1,m+1} \\ \beta_{j-1,m+2} \\ \vdots \end{Bmatrix} = \begin{bmatrix} \ddots & & & & & & & \\ & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 & 0 & 0 & \\ & 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 & \\ & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \\ & & & & & & & \ddots \end{bmatrix} \begin{Bmatrix} \vdots \\ \alpha_{j,2m} \\ \alpha_{j,2m+1} \\ \alpha_{j,2m+2} \\ \vdots \end{Bmatrix} \tag{3.86}$$

The block diagonal matrices that appear in equations (3.85) and (3.86) are referred to as decomposition filter matrices. For notational simplicity, let the decomposition filter matrices in equations (3.85) and (3.86) be denoted as $[A_{j-1,j}]$ and $[B_{j-1,j}]$, respectively, where the subscripts $j-1$ and j indicate between which two levels the decomposition is taking place. Introducing the vector notation:

$$\underline{\alpha}_j := \begin{Bmatrix} \vdots \\ \alpha_{j,2m} \\ \alpha_{j,2m+1} \\ \alpha_{j,2m+2} \\ \vdots \end{Bmatrix} \quad \underline{\alpha}_{j-1} := \begin{Bmatrix} \vdots \\ \alpha_{j-1,m} \\ \alpha_{j-1,m+1} \\ \alpha_{j-1,m+2} \\ \vdots \end{Bmatrix} \quad \underline{\beta}_{j-1} := \begin{Bmatrix} \vdots \\ \beta_{j-1,m} \\ \beta_{j-1,m+1} \\ \beta_{j-1,m+2} \\ \vdots \end{Bmatrix}$$

equations (3.85) and (3.86) can be written in the consolidated form

$$\underline{\alpha}_{j-1} = [A_{j-1,j}]\,\underline{\alpha}_j \tag{3.87}$$

$$\underline{\beta}_{j-1} = [B_{j-1,j}]\,\underline{\alpha}_j \tag{3.88}$$

The vector $\underline{\alpha}_j$ represents the scaling function coefficients on level j, and $\underline{\alpha}_{j-1}$, $\underline{\beta}_{j-1}$ are vectors composed of the scaling function and wavelet coefficients on level $j-1$. Note that, since level $j-1$ is twice as coarse as level j, the vectors $\underline{\alpha}_{j-1}$ and $\underline{\beta}_{j-1}$ contain half the number of elements as $\underline{\alpha}_j$. Because of the block diagonal form of $[A_{j-1,j}]$ and $[B_{j-1,j}]$, wavelet decomposition is computationally very efficient. In the next section, it will be shown that the decomposition filter matrices for all orthonormal wavelet families take the same general form as those for the Haar wavelet.

The inverse of the decomposition operation is known as reconstruction. It is not difficult to derive a reconstruction formula whereby the finer-scale coefficients $\{\alpha_{j,k}\}$ can be calculated from the coarser-scale coefficients $\{\alpha_{j-1,k}\}$ and $\{\beta_{j-1,k}\}$. This is accomplished by first taking the inner product of both sides of equation (3.81) with the scaling function $\phi_{j,m}$:

$$\begin{aligned}\sum_{k\in\mathbb{Z}} \alpha_{j,k} \int_{\mathbb{R}} \phi_{j,k}(x)\phi_{j,m}(x)dx &= \sum_{k\in\mathbb{Z}} \alpha_{j-1,k} \int_{\mathbb{R}} \phi_{j-1,k}(x)\phi_{j,m}(x)dx \\ &\quad + \sum_{k\in\mathbb{Z}} \beta_{j-1,k} \int_{\mathbb{R}} \psi_{j-1,k}(x)\phi_{j,m}(x)dx\end{aligned}$$

By orthogonality, this reduces to

$$\alpha_{j,m} = \sum_{k\in\mathbb{Z}} \alpha_{j-1,k} \int_{\mathbb{R}} \phi_{j-1,k}(x)\phi_{j,m}(x)dx + \sum_{k\in\mathbb{Z}} \beta_{j-1,k} \int_{\mathbb{R}} \psi_{j-1,k}(x)\phi_{j,m}(x)dx$$

Substituting for $\phi_{j-1,m}$ and $\psi_{j-1,m}$ using the two-scale equations, we obtain

$$\alpha_{j,m} = \sum_{k\in\mathbb{Z}} \alpha_{j-1,k} \int_{\mathbb{R}} \left[\tfrac{1}{\sqrt{2}}\phi_{j,2k}(x) + \tfrac{1}{\sqrt{2}}\phi_{j,2k+1}(x)\right] \phi_{j,m}(x)dx$$

$$+\sum_{k\in\mathbb{Z}}\beta_{j-1,k}\int_{\mathbb{R}}\left[\tfrac{1}{\sqrt{2}}\phi_{j,2k}(x)-\tfrac{1}{\sqrt{2}}\phi_{j,2k+1}(x)\right]\phi_{j,m}(x)dx$$

Finally, by the orthogonality of the scaling functions, the following reconstruction formula is obtained:

$$\alpha_{j,m}=\sum_k \alpha_{j-1,k}\left(\tfrac{1}{\sqrt{2}}\delta_{2k,m}+\tfrac{1}{\sqrt{2}}\delta_{2k+1,m}\right)+\sum_k \beta_{j-1,k}\left(\tfrac{1}{\sqrt{2}}\delta_{2k,m}-\tfrac{1}{\sqrt{2}}\delta_{2k+1,m}\right) \quad (3.89)$$

It is convenient to write the preceding reconstruction formula in matrix form:

$$\begin{Bmatrix}\vdots\\ \alpha_{j,m}\\ \alpha_{j,m+1}\\ \vdots\end{Bmatrix}=\begin{bmatrix}\ddots & & \\ \frac{1}{\sqrt{2}} & 0 & \\ \frac{1}{\sqrt{2}} & 0 & \\ 0 & \frac{1}{\sqrt{2}} & \\ 0 & \frac{1}{\sqrt{2}} & \\ & & \ddots\end{bmatrix}\begin{Bmatrix}\vdots\\ \alpha_{j-1,k}\\ \alpha_{j-1,k+1}\\ \vdots\end{Bmatrix}+\begin{bmatrix}\ddots & & \\ \frac{1}{\sqrt{2}} & 0 & \\ -\frac{1}{\sqrt{2}} & 0 & \\ 0 & \frac{1}{\sqrt{2}} & \\ 0 & -\frac{1}{\sqrt{2}} & \\ & & \ddots\end{bmatrix}\begin{Bmatrix}\vdots\\ \beta_{j-1,k}\\ \beta_{j-1,k+1}\\ \vdots\end{Bmatrix} \quad (3.90)$$

Using our vector notation and noting that the matrices that premultiply the vectors $\underline{\alpha}_{j-1}$ and $\underline{\beta}_{j-1}$ are the transposes of $[A_{j-1,j}]$ and $[B_{j-1,j}]$ respectively, equation (3.90) can be written as

$$\underline{\alpha}_j=[A_{j-1,j}]^T\,\underline{\alpha}_{j-1}+[B_{j-1,j}]^T\,\underline{\beta}_{j-1} \quad (3.91)$$

The decomposition filter matrices $[A_{j-1,j}]$ and $[B_{j-1,j}]$ are orthogonal transformations, meaning that they satisfy

$$[A_{j-1,j}]^T [A_{j-1,j}] = [I] \tag{3.92}$$

$$[B_{j-1,j}]^T [B_{j-1,j}] = [I] \tag{3.93}$$

where $[I]$ denotes the identity matrix.

Collectively, the decomposition and reconstruction formulas given by equations (3.83), (3.84), and (3.89) provide a fast, efficient means of calculating the scaling function and wavelet coefficients on one level from the scaling function coefficients on a level twice as fine, and vice versa. These equations are applied recursively to relate coefficients from nonadjacent levels, such as calculating the multiscale coefficients from the single-scale coefficients. While the results given in this section have been developed specifically for the Haar wavelet, it will be shown in the following section that they generalize to all orthonormal wavelet families.

3.4 Orthonormal Wavelets

In the preceding section, the two-scale equations were developed for the Haar scaling function and wavelet as a means of relating the scaling functions and wavelets on one level with those on an adjacent level. The multiresolution analysis that is generated by the Haar scaling function was discussed in detail. In particular, decomposition and reconstruction formulas were derived in order to relate the scaling function and wavelet coefficients on different levels. While derived specifically for the Haar wavelet family, these results can be generalized to all orthonormal wavelet families. That is, the foregoing results are applicable to all scaling function and wavelet pairs (ϕ, ψ) satisfying the following orthonormality conditions:

$$
\begin{aligned}
&(1) \qquad \langle \phi_{j,k}, \phi_{j,m} \rangle = \int_{\mathbb{R}} \phi_{j,k}(x)\phi_{j,m}(x)dx = \delta_{k,m} \\
&(2) \qquad \langle \psi_{j,k}, \psi_{l,m} \rangle = \int_{\mathbb{R}} \psi_{j,k}(x)\psi_{l,m}(x)dx = \delta_{(j,k),(l,m)} \\
&(3) \qquad \langle \phi_{j,k}, \psi_{l,m} \rangle = \int_{\mathbb{R}} \phi_{j,k}(x)\psi_{l,m}(x)dx = 0 \qquad \forall\ k, m \in \mathbb{Z},\ j \leq l
\end{aligned}
\tag{3.94}
$$

Clearly, the Haar wavelet is a specific example of an orthonormal wavelet.

Recall that the two-scale equations for the Haar scaling function and wavelet are given by

$$\phi_{j,k}(x) = \tfrac{1}{\sqrt{2}}\phi_{j+1,2k}(x) + \tfrac{1}{\sqrt{2}}\phi_{j+1,2k+1}(x) \tag{3.95}$$

$$\psi_{j,k}(x) = \tfrac{1}{\sqrt{2}}\phi_{j+1,2k}(x) - \tfrac{1}{\sqrt{2}}\phi_{j+1,2k+1}(x) \tag{3.96}$$

The two $\frac{1}{\sqrt{2}}$ factors that appear in equation (3.95) and the $\frac{1}{\sqrt{2}}$ and $-\frac{1}{\sqrt{2}}$ factors that appear in equation (3.96) are termed scaling function and wavelet filters, respectively. If we define

$$
\begin{aligned}
\{a_s\} &= \{a_0, a_1\} = \left\{\tfrac{1}{\sqrt{2}}, \tfrac{1}{\sqrt{2}}\right\} \\
\{b_s\} &= \{b_0, b_1\} = \left\{\tfrac{1}{\sqrt{2}}, -\tfrac{1}{\sqrt{2}}\right\}
\end{aligned}
$$

where $\{a_s\}$ represents the scaling function filters and $\{b_s\}$ represents the wavelet filters, equations (3.95) and (3.96) can be written in the form

$$\phi_{j,k}(x) = \sum_s a_s \phi_{j+1,2k+s}(x) \tag{3.97}$$

$$\psi_{j,k}(x) = \sum_s b_s \phi_{j+1,2k+s}(x) \tag{3.98}$$

In general, every scaling function and wavelet pair has a unique set of filters $\{a_s\}$ and $\{b_s\}$. The number and range of filter coefficients in $\{a_s\}$ and $\{b_s\}$ differs for each pair and depends on the support of the functions ϕ and ψ. The two-scale relationships for all orthonormal wavelet families take the form shown in equations (3.97) and (3.98).

Multiresolution analyses are generated from orthonormal scaling functions in exactly the same manner as the Haar scaling function. For a given orthonormal scaling function and wavelet pair (ϕ, ψ), each space V_j is defined as

$$V_j := span\{\phi_{j,k}\}_{k\in\mathbb{Z}} \tag{3.99}$$

where

$$\phi_{j,k}(x) = 2^{j/2}\phi\left(2^j x - k\right) \tag{3.100}$$

Each wavelet complement space W_j is defined as

$$W_j := span\{\psi_{j,k}\}_{k\in\mathbb{Z}} \tag{3.101}$$

where

$$\psi_{j,k}(x) = 2^{j/2}\psi\left(2^j x - k\right) \tag{3.102}$$

Recall that the space V_j can be decomposed in terms of the coarser-scale spaces V_{j-1} and W_{j-1}:

$$V_j = V_{j-1} \oplus W_{j-1} \tag{3.103}$$

Applying equation (3.103) recursively, the following multilevel decomposition of V_j is obtained:

$$V_j = V_{j_0} \oplus W_{j_0} \oplus W_{j_0-1} \oplus \cdots \oplus W_{j-2} \oplus W_{j-1} \tag{3.104}$$

Recall that the starting point of a multiresolution analysis of a function or signal f is a single-scale approximation of the form

$$f_j(x) = \sum_{k\in\mathbb{Z}} \alpha_{j,k}\phi_{j,k}(x) \tag{3.105}$$

which is in terms of the scaling functions that span a fine-scale space V_j. Equation (3.103) leads to an equivalent, two-scale expansion:

$$f_j(x) = \sum_{k\in\mathbb{Z}} \alpha_{j-1,k}\phi_{j-1,k}(x) + \sum_{k\in\mathbb{Z}} \beta_{j-1,k}\psi_{j-1,k}(x) \tag{3.106}$$

Finally, equation (3.104) implies that we have a multilevel representation of f_j of the form

$$f_j(x) = \sum_{k\in\mathbb{Z}} \alpha_{j_0,k}\phi_{j_0,k}(x) + \sum_{l=j_0}^{j-1}\sum_{k\in\mathbb{Z}} \beta_{l,k}\psi_{l,k}(x) \tag{3.107}$$

which is in terms of the wavelets supported on levels j_0 through $j-1$ and the scaling functions on only the coarsest level j_0.

Recall that, in the previous section, decomposition formulas were developed for the Haar wavelet. Applied recursively, these equations provide a fast, efficient tool for calculating the coefficients that appear in the multilevel representation of a function. A reconstruction formula, which performs the inverse operation to decomposition, was also derived. The derivation of generalized decomposition and reconstruction formulas for orthonormal wavelet families follows very closely from the development for the Haar wavelet. In order to derive these formulas for a given orthonormal scaling function and wavelet pair (ϕ, ψ), we first equate the single-scale and two-scale representations of a function f. From equations (3.105) and (3.106), we have

$$\sum_{k\in\mathbb{Z}} \alpha_{j,k}\phi_{j,k}(x) = \sum_{k\in\mathbb{Z}} \alpha_{j-1,k}\phi_{j-1,k}(x) + \sum_{k\in\mathbb{Z}} \beta_{j-1,k}\psi_{j-1,k}(x) \tag{3.108}$$

A decomposition formula for the coefficients $\{\alpha_{j-1,k}\}$ can be obtained by taking the $L^2(\mathbb{R})$ inner product of both sides of equation (3.108) with the scaling function $\phi_{j-1,m}$:

$$\begin{aligned}\sum_k \alpha_{j,k} \int_{\mathbb{R}} \phi_{j,k}(x)\phi_{j-1,m}(x)dx &= \sum_k \alpha_{j-1,k} \int_{\mathbb{R}} \phi_{j-1,k}(x)\phi_{j-1,m}(x)dx \\ &\quad + \sum_k \beta_{j-1,k} \int_{\mathbb{R}} \psi_{j-1,k}(x)\phi_{j-1,m}(x)dx\end{aligned}$$

Using the orthogonality of the scaling functions and wavelets, this reduces to

$$\alpha_{j-1,m} = \sum_{k\in\mathbb{Z}} \alpha_{j,k} \int_{\mathbb{R}} \phi_{j,k}(x)\phi_{j-1,m}(x)dx$$

Substituting for $\phi_{j-1,m}$ using the two-scale equation, equation (3.97), we have

$$\alpha_{j-1,m} = \sum_k \alpha_{j,k} \int_{\mathbb{R}} \sum_s a_s \phi_{j,2m+s}(x)\phi_{j,k}(x)dx$$

By orthogonality, this can be simplified to

$$\alpha_{j-1,m} = \sum_{k,s} \alpha_{j,k} a_s \delta_{k,2m+s}$$

Finally, the following decomposition formula for $\alpha_{j-1,m}$ is obtained:

$$\alpha_{j-1,m} = \sum_k \alpha_{j,k} a_{k-2m} \tag{3.109}$$

Similarly, a decomposition formula for $\beta_{j-1,m}$ is given by

$$\beta_{j-1,m} = \sum_k \alpha_{j,k} b_{k-2m} \tag{3.110}$$

Equation (3.110) can be derived by taking the inner product of both sides of equation (3.108) with $\psi_{j-1,m}$, an arbitrary wavelet on level $j-1$. The two-scale relationship

in equation (3.98) and the orthogonality properties are then used to simplify the resulting expression to the form shown in equation (3.110).

It is convenient to express the results from equations (3.109) and (3.110) in matrix form:

$$\begin{Bmatrix} \vdots \\ \alpha_{j-1,k} \\ \alpha_{j-1,k+1} \\ \alpha_{j-1,k+2} \\ \vdots \end{Bmatrix} = \begin{bmatrix} \ddots & & & & & & & \\ & a_0 & a_1 & a_2 & a_3 & 0 & 0 & \\ & 0 & 0 & a_0 & a_1 & a_2 & a_3 & \\ & 0 & 0 & 0 & 0 & a_0 & a_1 & \\ & & & & & & & \ddots \end{bmatrix} \begin{Bmatrix} \vdots \\ \alpha_{j,2k} \\ \alpha_{j,2k+1} \\ \alpha_{j,2k+2} \\ \vdots \end{Bmatrix} \tag{3.111}$$

$$\begin{Bmatrix} \vdots \\ \beta_{j-1,k} \\ \beta_{j-1,k+1} \\ \beta_{j-1,k+2} \\ \vdots \end{Bmatrix} = \begin{bmatrix} \ddots & & & & & & & \\ & b_0 & b_1 & b_2 & b_3 & 0 & 0 & \\ & 0 & 0 & b_0 & b_1 & b_2 & b_3 & \\ & 0 & 0 & 0 & 0 & b_0 & b_1 & \\ & & & & & & & \ddots \end{bmatrix} \begin{Bmatrix} \vdots \\ \alpha_{j,2k} \\ \alpha_{j,2k+1} \\ \alpha_{j,2k+2} \\ \vdots \end{Bmatrix} \tag{3.112}$$

Once again, note that the number of filter coefficients and the range of indices is dependent on the support of the specific scaling function and wavelet pair being used. As an example, equations (3.111) and (3.112) show the case where there are four scaling function and wavelet filter coefficients with indices ranging from 0 to 3. Following the same notation as before, let the decomposition filter matrices in equations (3.111) and (3.112) be denoted as $[A_{j-1,j}]$ and $[B_{j-1,j}]$, respectively. Using vector notation, equations (3.111) and (3.112) can be written as

$$\underline{\alpha}_{j-1} = [A_{j-1,j}]\,\underline{\alpha}_j \tag{3.113}$$

$$\underline{\beta}_{j-1} = [B_{j-1,j}]\,\underline{\alpha}_j \tag{3.114}$$

where $\underline{\alpha}_j$ represents a vector of the scaling function coefficients on level j and $\underline{\alpha}_{j-1}$, $\underline{\beta}_{j-1}$ are vectors representing the scaling function and wavelet coefficients on level $j-1$.

In general, a multiresolution analysis is carried out over a number of levels. For example, suppose a function is to be decomposed over two levels. It is known, from equations (3.113) and (3.114), that

$$\underline{\alpha}_{j-1} = [A_{j-1,j}]\,\underline{\alpha}_j$$

$$\underline{\beta}_{j-1} = [B_{j-1,j}]\,\underline{\alpha}_j$$

Similarly, we can write

$$\underline{\alpha}_{j-2} = [A_{j-2,j-1}]\,\underline{\alpha}_{j-1}$$

$$\underline{\beta}_{j-2} = [B_{j-2,j-1}]\,\underline{\alpha}_{j-1}$$

Often, $\underline{\alpha}_{j-1}$ and $\underline{\beta}_{j-1}$ are combined into a stacked vector so that equations (3.113) and (3.114) take the form

$$\left\{ \begin{array}{c} \underline{\alpha}_{j-1} \\ - - - \\ \underline{\beta}_{j-1} \end{array} \right\} = \left[\begin{array}{c} [A_{j-1,j}] \\ - - - \\ [B_{j-1,j}] \end{array} \right] \underline{\alpha}_j \tag{3.115}$$

Since $\underline{\alpha}_{j-2}$ and $\underline{\beta}_{j-2}$ are calculated from $\underline{\alpha}_{j-1}$ and are not dependent on $\underline{\beta}_{j-1}$, we can write

$$\left\{\begin{array}{c} \underline{\alpha}_{j-2} \\ - - - \\ \underline{\beta}_{j-2} \end{array}\right\} = \left[\begin{array}{c|c} [A_{j-2,j-1}] & 0 \\ - - - & - - - \\ [B_{j-2,j-1}] & 0 \end{array}\right] \left\{\begin{array}{c} \underline{\alpha}_{j-1} \\ - - - \\ \underline{\beta}_{j-1} \end{array}\right\} \tag{3.116}$$

Combining equations (3.115) and (3.116), the following expression is obtained:

$$\underline{\beta} = [T_{j-2,j}]\underline{\alpha}_j \tag{3.117}$$

where the vector of multiscale coefficients $\underline{\beta}$ and the transformation matrix $[T_{j-2,j}]$ are defined as

$$\underline{\beta} := \left\{\begin{array}{c} \underline{\alpha}_{j-2} \\ - - - \\ \underline{\beta}_{j-2} \\ - - - \\ \\ \underline{\beta}_{j-1} \end{array}\right\} \tag{3.118}$$

$$[T_{j-2,j}] := \left[\begin{array}{c|c|c} [A_{j-2,j-1}] & & \\ - - - & 0 & [A_{j-1,j}] \\ [B_{j-2,j-1}] & & \\ - - - & - - - & - - - - - \\ & & \\ 0 & [I] & [B_{j-1,j}] \\ & & \end{array}\right] \tag{3.119}$$

Recall that in a wavelet-based multiresolution analysis, a function is expressed as a linear combination of wavelets over all the levels used in the decomposition and the scaling functions on only the coarsest level. In the preceding example, where

the function is decomposed over two levels, the vector of multiscale coefficients is composed of wavelet coefficients on both levels, $\underline{\beta}_{j-1}$ and $\underline{\beta}_{j-2}$, and scaling coefficients $\underline{\alpha}_{j-2}$ on the coarsest level alone. Equation (3.117) shows in matrix form how the multiscale coefficients are obtained from the single-scale coefficients $\underline{\alpha}_j$ for the example we have considered. This result can easily be extended to decompositions over more than two levels. It should be noted that, in practice, one does not compute the wavelet transform by explicitly forming a decomposition matrix such as the one shown in equation (3.117). Instead, the decomposition formulas given by equations (3.109) and (3.110) are applied recursively. The matrix form has been given here to provide additional insight into the structure of decomposition over several levels.

A reconstruction formula, applicable to all orthonormal wavelet families, can be derived by taking the inner product of equation (3.108) with the scaling function $\phi_{j,m}$:

$$\begin{aligned}\sum_k \alpha_{j,k} \int_{\mathbb{R}} \phi_{j,k}(x)\phi_{j,m}(x)dx &= \sum_k \alpha_{j-1,k} \int_{\mathbb{R}} \phi_{j-1,k}(x)\phi_{j,m}(x)dx \\ &\quad + \sum_k \beta_{j-1,k} \int_{\mathbb{R}} \psi_{j-1,k}(x)\phi_{j,m}(x)dx\end{aligned}$$

By orthogonality, this reduces to

$$\alpha_{j,m} = \sum_k \alpha_{j-1,k} \int_{\mathbb{R}} \phi_{j-1,k}(x)\phi_{j,m}(x)dx + \sum_k \beta_{j-1,k} \int_{\mathbb{R}} \psi_{j-1,k}(x)\phi_{j,m}(x)dx$$

Substituting for $\phi_{j-1,k}$ and $\psi_{j-1,k}$ in terms of the two-scale equations, we obtain

$$\begin{aligned}\alpha_{j,m} &= \sum_k \alpha_{j-1,k} \int_{\mathbb{R}} \sum_s a_s \phi_{j,2k+s}\phi_{j,m}(x)dx \\ &\quad + \sum_k \beta_{j-1,k} \int_{\mathbb{R}} \sum_s b_s \phi_{j,2k+s}\phi_{j,m}(x)dx\end{aligned}$$

Making use of the orthogonality of the scaling functions, this can be simplified to the form

$$\alpha_{j,m} = \sum_{k,s} \alpha_{j-1,k} a_s \delta_{2k+s,m} + \sum_{k,s} \beta_{j-1,k} b_s \delta_{2k+s,m}$$

Finally, the following reconstruction formula for $\alpha_{j,m}$ is obtained:

$$\alpha_{j,m} = \sum_{k} \alpha_{j-1,k} a_{m-2k} + \sum_{k} \beta_{j-1,k} b_{m-2k} \tag{3.120}$$

It is convenient to write equation (3.120) in matrix form:

$$\begin{Bmatrix} \vdots \\ \alpha_{j,k} \\ \alpha_{j,k+1} \\ \vdots \end{Bmatrix} = \begin{bmatrix} \ddots & \\ a_0 & 0 \\ a_1 & 0 \\ a_2 & a_0 \\ a_3 & a_1 \\ 0 & a_2 \\ 0 & a_3 \\ & \ddots \end{bmatrix} \begin{Bmatrix} \vdots \\ \alpha_{j-1,k} \\ \alpha_{j-1,k+1} \\ \vdots \end{Bmatrix} + \begin{bmatrix} \ddots & \\ b_0 & 0 \\ b_1 & 0 \\ b_2 & b_0 \\ b_3 & b_1 \\ 0 & b_2 \\ 0 & b_3 \\ & \ddots \end{bmatrix} \begin{Bmatrix} \vdots \\ \beta_{j-1,k} \\ \beta_{j-1,k+1} \\ \vdots \end{Bmatrix} \tag{3.121}$$

Just as in the Haar case, the matrices in the reconstruction formula are the transposes of the decomposition filter matrices $[A_{j-1,j}]$ and $[B_{j-1,j}]$, and equation (3.121) can be written as

$$\underline{\alpha}_j = [A_{j-1,j}]^T \underline{\alpha}_{j-1} + [B_{j-1,j}]^T \underline{\beta}_{j-1} \tag{3.122}$$

In general, wavelet reconstruction is carried out over multiple levels. Consider an example in which the reconstruction is taking place over two levels, starting at level $j-2$. This implies that the vectors of coefficients $\underline{\beta}_{j-1}$, $\underline{\beta}_{j-2}$, and $\underline{\alpha}_{j-2}$ are all known. The goal of the reconstruction, then, is to calculate the single-scale coefficients $\underline{\alpha}_j$. It is clear from equation (3.123) that we can write

$$\underline{\alpha}_{j-1} = [A_{j-2,j-1}]^T \underline{\alpha}_{j-2} + [B_{j-2,j-1}]^T \underline{\beta}_{j-2} \tag{3.123}$$

Equations (3.123) and (3.121) can be written in the form

$$\underline{\alpha}_{j-1} = \left[\begin{array}{c|c} [A_{j-2,j-1}]^T & [B_{j-2,j-1}]^T \end{array}\right] \left\{\begin{array}{c} \underline{\alpha}_{j-2} \\ --- \\ \underline{\beta}_{j-2} \end{array}\right\} \tag{3.124}$$

$$\underline{\alpha}_j = \left[\begin{array}{c|c} [A_{j-1,j}]^T & [B_{j-1,j}]^T \end{array}\right] \left\{\begin{array}{c} \underline{\alpha}_{j-1} \\ --- \\ \underline{\beta}_{j-1} \end{array}\right\} \tag{3.125}$$

Combining equations (3.124) and (3.125), we obtain

$$\underline{\alpha}_j = [T_{j,j-2}]\underline{\beta} \tag{3.126}$$

where $\underline{\beta}$ is the multiscale vector defined in equation (3.118) and $[T_{j,j-2}]$ is defined as

$$[T_{j,j-2}] := \left[\begin{array}{c|c} & \\ [A_{j-1,j}]^T & [B_{j-1,j}]^T \\ & \end{array}\right] \left[\begin{array}{c|c|c} [A_{j-2,j-1}]^T & [B_{j-2,j-1}]^T & 0 \\ \hline 0 & 0 & [I] \end{array}\right] \tag{3.127}$$

Note that, comparing equation (3.127) with equation (3.119), we have

$$[T_{j,j-2}] = [T_{j-2,j}]^T \tag{3.128}$$

That is, $[T_{j-2,j}]$, and the wavelet transform in general, is an orthogonal transformation.

3.5 Orthonormal Multiwavelets

In the previous section, the Haar wavelet discussion was extended to include all orthonormal wavelet families. In this section, the discussion will be generalized further to consider multiresolution analyses that are generated by several scaling functions $\{\phi^1, \ldots, \phi^r\}$, where r is the number of generators. The r associated wavelets $\{\psi^1, \ldots, \psi^r\}$ are termed multiwavelets. The case where $r = 1$, which has already been discussed in detail, is known as the single-generator case. The motivation for multiwavelets is that using several functions enables more freedom in designing wavelet families. For example, it is possible to construct piecewise-polynomial multiwavelets that are orthonormal, compactly-supported, and have arbitrary approximation order. In addition, one can obtain analytical forms for many multiwavelets. In the next chapter, multiwavelets that satisfy these properties will be constructed from the classical finite element basis functions. It will be demonstrated that this particular

class of multiwavelets can easily be adapted to boundaries. This section focuses on developing general results for orthonormal multiwavelets and their associated multiresolution analyses.

Given r scaling functions $\{\phi^1, \ldots, \phi^r\}$ and r associated wavelets $\{\psi^1, \ldots, \psi^r\}$, the following vectors can be defined:

$$\underline{\Phi} := \begin{Bmatrix} \phi^1 \\ \phi^2 \\ \vdots \\ \phi^r \end{Bmatrix} \qquad \underline{\Psi} := \begin{Bmatrix} \psi^1 \\ \psi^2 \\ \vdots \\ \psi^r \end{Bmatrix} \tag{3.129}$$

Orthonormal multiwavelets satisfy the following orthogonality conditions:

$$\begin{aligned} &(1) \qquad \int_{\mathbb{R}} \phi^s_{j,k}(x)\phi^t_{j,m}(x)dx = \delta_{(k,s),(m,t)} \\ &(2) \qquad \int_{\mathbb{R}} \psi^s_{j,k}(x)\psi^t_{l,m}(x)dx = \delta_{(j,k,s),(l,m,t)} \\ &(3) \qquad \int_{\mathbb{R}} \phi^s_{j,k}(x)\psi^t_{l,m}(x)dx = 0 \quad \forall\ k,m \in \mathbb{Z}, \quad s,t \in \{1,\ldots,r\}, \quad j \le l \end{aligned} \tag{3.130}$$

where we have the notation

$$\phi^s_{j,k} := 2^{j/2}\phi^s\left(2^j x - k\right) \tag{3.131}$$

$$\psi^s_{j,k} := 2^{j/2}\psi^s\left(2^j x - k\right) \tag{3.132}$$

Using the vector notation in equation (3.129), the orthogonality properties in equation (3.130) can be written in the equivalent form

$$(1) \qquad \int_{\mathbb{R}} \underline{\Phi}_{j,k}(x)\underline{\Phi}^T_{j,m}(x)dx = [I]\,\delta_{k,m}$$

$$(2) \qquad \int_{\mathbb{R}} \underline{\Psi}_{j,k}(x)\underline{\Psi}^T_{l,m}(x)dx = [I]\,\delta_{(j,k),(l,m)} \tag{3.133}$$

$$(3) \qquad \int_{\mathbb{R}} \underline{\Phi}_{j,k}(x)\underline{\Psi}^T_{l,m}(x)dx = [0] \quad \forall\ k,m \in \mathbb{Z},\ \ j \le l$$

$$(4) \qquad \int_{\mathbb{R}} \underline{\Psi}_{j,k}(x)\underline{\Phi}^T_{l,m}(x)dx = [0] \quad \forall\ k,m \in \mathbb{Z},\ \ j \ge l$$

where $[I]$ represents an r x r identity matrix and $[0]$ denotes an r x r matrix of zeros. The two-scale equations for multiwavelet families are given by

$$\phi^s_{j-1,k}(x) = \sum_{p,t} a^{s,t}_p \phi^t_{j,2k+p}(x) \tag{3.134}$$

$$\psi^s_{j-1,k}(x) = \sum_{p,t} b^{s,t}_p \phi^t_{j,2k+p}(x) \tag{3.135}$$

In vector form, the two-scale equations can be written as

$$\underline{\Phi}_{j-1,k}(x) = \sum_{p} [a_p]\,\underline{\Phi}_{j,2k+p}(x) \tag{3.136}$$

$$\underline{\Psi}_{j-1,k}(x) = \sum_{p} [b_p]\,\underline{\Phi}_{j,2k+p}(x) \tag{3.137}$$

Note that these two-scale equations take the same form as in the single-generator case. The main difference is that the scaling function and wavelet filters $\{[a_p]\}$, $\{[b_p]\}$ are matrices instead of scalars.

A multiresolution analysis that is generated by r scaling functions $\{\phi^1, \ldots, \phi^r\}$ is termed a multiresolution analysis of multiplicity r. A multiresolution analysis of multiplicity r is a family of subspaces $\{V_j\}_{j\in\mathbb{Z}}$ of $L^2(\mathbb{R})$ satisfying the following properties [2]:

$$(1) \qquad \cdots V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \cdots$$

$$
\begin{aligned}
&(2) \quad \bigcap_{j\in\mathbb{Z}} V_j = \{0\} \\
&(3) \quad \overline{\bigcup_{j\in\mathbb{Z}} V_j} = L^2(\mathbb{R}) \\
&(4) \quad f(x) \in V_j \Longleftrightarrow f(2x) \in V_{j+1} \\
&(5) \quad f(x) \in V_j \Longrightarrow f(x-k) \in V_j \quad \forall\, k \in \mathbb{Z} \\
&(6) \quad \text{There are } r \text{ functions } \left\{\phi^1, \ldots, \phi^r\right\} \text{ such that the set of translates} \\
&\qquad \left\{\phi^s(x-k): \quad s = 1, \ldots, r\right\}_{k\in\mathbb{Z}} \text{ forms a Riesz basis for } V_0
\end{aligned}
\tag{3.138}
$$

The properties of a multiresolution analysis that is generated by multiple scaling functions are essentially the same as those for the single-generator case. The only difference is in the number of generators. The space V_0 is defined as

$$V_0 := span\left\{\phi^s(\cdot - k): \quad s = 1, \ldots, r\right\}_{k\in\mathbb{Z}} \tag{3.139}$$

and, in general, the space V_j is defined as

$$V_j := span\left\{\phi^s_{j,k}: \quad s = 1, \ldots, r\right\}_{k\in\mathbb{Z}} \tag{3.140}$$

Similarly, the wavelet space W_j that comprises the difference between two adjacent spaces V_{j+1} and V_j:

$$W_j = V_{j+1} \ominus V_j \tag{3.141}$$

is defined as

$$W_j := span\left\{\psi^s_{j,k}: \quad s = 1, \ldots, r\right\}_{k\in\mathbb{Z}} \tag{3.142}$$

The space W_0 is defined as

$$W_0 := span\left\{\psi^s(\cdot - k): \quad s = 1, \ldots, r\right\}_{k\in\mathbb{Z}} \tag{3.143}$$

Just as before, we have the following multiscale decomposition of V_j:

$$V_j = V_{j_0} \oplus W_{j_0} \oplus W_{j_0-1} \oplus \cdots \oplus W_{j-2} \oplus W_{j-1} \tag{3.144}$$

where j_0 is the coarsest level in the decomposition.

Similar to the single-generator case, we have a single-scale approximation of a function f of the form

$$f_j(x) = \sum_{s=1}^{r} \sum_{k\in\mathbb{Z}} \alpha_{j,k}^s \phi_{j,k}^s(x) \tag{3.145}$$

and an equivalent, two-scale expansion that can be written as

$$f_j(x) = \sum_{s=1}^{r} \sum_{k\in\mathbb{Z}} \alpha_{j-1,k}^s \phi_{j-1,k}^s(x) + \sum_{s=1}^{r} \sum_{k\in\mathbb{Z}} \beta_{j-1,k}^s \psi_{j-1,k}^s(x) \tag{3.146}$$

A multilevel representation of f_j corresponding to equation (3.144) is given by

$$f_j(x) = \sum_{s=1}^{r} \sum_{k\in\mathbb{Z}} \alpha_{j_0,k}^s \phi_{j_0,k}^s(x) + \sum_{l=j_0}^{j-1} \sum_{s=1}^{r} \sum_{k\in\mathbb{Z}} \beta_{l,k}^s \psi_{l,k}^s(x) \tag{3.147}$$

Note that the expressions for f_j in equations (3.145) through (3.147) are of the same form as the expansions that are given in the single-generator case. The only difference is that, for multiwavelets, it is necessary to sum over the number of different functions as well. Of course, when $r = 1$, equations (3.145) through (3.147) reduce to the single-generator results. Introducing the vector notation

$$\underline{\alpha}_{j,k} := \begin{Bmatrix} \alpha_{j,k}^1 \\ \vdots \\ \alpha_{j,k}^r \end{Bmatrix} \tag{3.148}$$

the single-scale approximation in equation (3.145) can be written as

$$f_j(x) = \sum_{k\in\mathbb{Z}} \underline{\alpha}_{j,k}^T \underline{\Phi}_{j,k}(x) \tag{3.149}$$

and the two-scale expansion in equation (3.146) takes the form

$$f_j(x) = \sum_{k\in\mathbb{Z}} \underline{\alpha}_{j-1,k}^T \underline{\Phi}_{j-1,k}(x) + \sum_{k\in\mathbb{Z}} \underline{\beta}_{j-1,k}^T \underline{\Psi}_{j-1,k}(x) \tag{3.150}$$

The derivation of decomposition formulas which can be used recursively to calculate the multiscale coefficients in equation (3.147) follows the same procedure as the single-generator case. It is convenient to work with the vector form of the equations, although, as will be seen later, the vector notation is not easy to generalize to higher dimensions. The first step in deriving multiwavelet decomposition formulas is to express the equivalence of the single-scale and two-scale expansions of a function:

$$\sum_{k\in\mathbb{Z}} \underline{\alpha}_{j,k}^T \underline{\Phi}_{j,k}(x) = \sum_{k\in\mathbb{Z}} \underline{\alpha}_{j-1,k}^T \underline{\Phi}_{j-1,k}(x) + \sum_{k\in\mathbb{Z}} \underline{\beta}_{j-1,k}^T \underline{\Psi}_{j-1,k}(x) \tag{3.151}$$

Post-multiplying both sides of equation (3.151) by $\underline{\Phi}_{j-1,m}^T$ and integrating over $\mathbb{R}$, we have

$$\begin{aligned}\sum_k \underline{\alpha}_{j,k}^T \int_{\mathbb{R}} \underline{\Phi}_{j,k}(x)\underline{\Phi}_{j-1,m}^T(x)dx &= \sum_k \underline{\alpha}_{j-1,k}^T \int_{\mathbb{R}} \underline{\Phi}_{j-1,k}(x)\underline{\Phi}_{j-1,m}^T(x)dx \\ &\quad + \sum_k \underline{\beta}_{j-1,k}^T \int_{\mathbb{R}} \underline{\Psi}_{j-1,k}(x)\underline{\Phi}_{j-1,m}^T(x)dx\end{aligned}$$

By orthogonality, this reduces to

$$\sum_k \underline{\alpha}_{j,k}^T \int_{\mathbb{R}} \underline{\Phi}_{j,k}(x)\underline{\Phi}_{j-1,m}^T(x)dx = \sum_k \underline{\alpha}_{j-1,k}^T [I]\, \delta_{k,m}$$

and

$$\underline{\alpha}_{j-1,m}^T = \sum_k \underline{\alpha}_{j,k}^T \int_{\mathbb{R}} \underline{\Phi}_{j,k}(x)\underline{\Phi}_{j-1,m}^T(x)dx$$

Transposing the two-scale relationship, equation (3.136), we obtain the following expression for $\underline{\Phi}_{j-1,m}^T$:

$$\underline{\Phi}_{j-1,m}^T(x) = \sum_p \underline{\Phi}_{j,2m+p}^T(x) \left[a_p\right]^T$$

Then, substituting for $\underline{\Phi}_{j-1,m}^T$, we have

$$\underline{\alpha}_{j-1,m}^T = \sum_{k,p} \underline{\alpha}_{j,k}^T \int_{\mathbb{R}} \underline{\Phi}_{j,k}(x) \left(\underline{\Phi}_{j,2m+p}^T(x) \left[a_p\right]^T\right) dx$$

Once again using the orthogonality properties of the multiwavelets, this becomes

$$\underline{\alpha}_{j-1,m}^T = \sum_{k,p} \underline{\alpha}_{j,k}^T \left[I\right] \delta_{k,2m+p} \left[a_p\right]^T$$

Finally, this can be simplified to

$$\underline{\alpha}_{j-1,m}^T = \sum_k \underline{\alpha}_{j,k}^T \left[a_{k-2m}\right]^T$$

Taking the transpose, the following decomposition formula is obtained:

$$\underline{\alpha}_{j-1,m} = \sum_k \left[a_{k-2m}\right] \underline{\alpha}_{j,k} \tag{3.152}$$

Following a similar procedure, the following decomposition formula for the vectors of wavelet coefficients $\{\underline{\beta}_{j-1,m}\}_{m\in\mathbb{Z}}$ can be derived:

$$\underline{\beta}_{j-1,m} = \sum_k \left[b_{k-2m}\right] \underline{\alpha}_{j,k} \tag{3.153}$$

Equation (3.153) is obtained by post-multiplying equation (3.151) by $\underline{\Psi}_{j-1,m}^T$ and integrating over $\mathbb{R}$. The resulting expression is then simplified by using the orthogonality properties of the multiwavelets and the two-scale relationship given in equation (3.136). Note that the decomposition formulas can also be written in the form

$$\alpha_{j-1,m}^s = \sum_{t=1}^r \sum_k a_{k-2m}^{s,t} \alpha_{j,k}^t \tag{3.154}$$

$$\beta_{j-1,m}^{s} = \sum_{t=1}^{r} \sum_{k} b_{k-2m}^{s,t} \alpha_{j,k}^{t} \tag{3.155}$$

A reconstruction formula can be derived for orthonormal multiwavelet families by post-multiplying equation (3.151) by $\underline{\Phi}_{j,m}^{T}$ and integrating over $\mathbb{R}$:

$$\begin{aligned} \sum_{k} \underline{\alpha}_{j,k}^{T} \int_{\mathbb{R}} \underline{\Phi}_{j,k}(x) \underline{\Phi}_{j,m}^{T}(x) dx &= \sum_{k} \underline{\alpha}_{j-1,k}^{T} \int_{\mathbb{R}} \underline{\Phi}_{j-1,k}(x) \underline{\Phi}_{j,m}^{T}(x) dx \\ &\quad + \sum_{k} \underline{\beta}_{j-1,k}^{T} \int_{\mathbb{R}} \underline{\Psi}_{j-1,k}(x) \underline{\Phi}_{j,m}^{T}(x) dx \end{aligned}$$

Using the orthogonality properties of the multiwavelets, we obtain

$$\begin{aligned} \sum_{k} \underline{\alpha}_{j,k}^{T} [I] \, \delta_{k,m} &= \sum_{k} \underline{\alpha}_{j-1,k}^{T} \int_{\mathbb{R}} \underline{\Phi}_{j-1,k}(x) \underline{\Phi}_{j,m}^{T}(x) dx \\ &\quad + \sum_{k} \underline{\beta}_{j-1,k}^{T} \int_{\mathbb{R}} \underline{\Psi}_{j-1,k}(x) \underline{\Phi}_{j,m}^{T}(x) dx \end{aligned}$$

and

$$\underline{\alpha}_{j,m}^{T} = \sum_{k} \underline{\alpha}_{j-1,k}^{T} \int_{\mathbb{R}} \underline{\Phi}_{j-1,k}(x) \underline{\Phi}_{j,m}^{T}(x) dx + \sum_{k} \underline{\beta}_{j-1,k}^{T} \int_{\mathbb{R}} \underline{\Psi}_{j-1,k}(x) \underline{\Phi}_{j,m}^{T}(x) dx$$

Substituting for $\underline{\Phi}_{j-1,k}^{T}$ and $\underline{\Psi}_{j-1,k}^{T}$ using the two-scale equations, we have

$$\begin{aligned} \underline{\alpha}_{j,m}^{T} &= \sum_{k,p} \underline{\alpha}_{j-1,k}^{T} \int_{\mathbb{R}} [a_p] \, \underline{\Phi}_{j,2k+p}(x) \underline{\Phi}_{j,m}^{T}(x) dx \\ &\quad + \sum_{k,p} \underline{\beta}_{j-1,k}^{T} \int_{\mathbb{R}} [b_p] \, \underline{\Phi}_{j,2k+p}(x) \underline{\Phi}_{j,m}^{T}(x) dx \end{aligned}$$

By orthogonality, this reduces to

$$\underline{\alpha}_{j,m}^T = \sum_{k,p} \underline{\alpha}_{j-1,k}^T [a_p] [I] \delta_{m,2k+p} + \sum_{k,p} \underline{\beta}_{j-1,k}^T [b_p] [I] \delta_{m,2k+p}$$

Finally, this can be simplified to the form

$$\underline{\alpha}_{j,m}^T = \sum_k \underline{\alpha}_{j-1,k}^T [a_{m-2k}] + \sum_k \underline{\beta}_{j-1,k}^T [b_{m-2k}]$$

Transposing this result, the following reconstruction formula is obtained:

$$\underline{\alpha}_{j,m} = \sum_k \left\{ [a_{m-2k}]^T \underline{\alpha}_{j-1,k} + [b_{m-2k}]^T \underline{\beta}_{j-1,k} \right\} \tag{3.156}$$

Note that equation (3.156) can be written in the equivalent form

$$\alpha_{j,m}^s = \sum_{t=1}^{r} \sum_k \left\{ a_{m-2k}^{t,s} \alpha_{j-1,k}^t + b_{m-2k}^{t,s} \beta_{j-1,k}^t \right\} \tag{3.157}$$

3.6 Tensor Product Wavelets

To this point, the discussion has been focused on wavelets in one dimension. For many wavelet families, the preceding results can easily be extended to higher dimensions using tensor product wavelets. In this section, two-dimensional tensor product wavelets and their associated multiresolution analyses will be discussed in detail. In particular, decomposition and reconstruction formulas, similar to those derived for the one-dimensional case, will be developed. The discussion will treat the more general case of tensor product multiwavelets, since single-generator wavelet families can be viewed as specific types of multiwavelets.

Given a multiresolution analysis $\{V_j\}_{j \in \mathbb{Z}}$ of multiplicity r, generated by r scaling functions $\{\phi^1, \ldots, \phi^r\}$, two-dimensional scaling functions can be obtained by taking the tensor products of the scaling functions in one dimension (x-direction) with those in the other dimension (y-direction). Hence, a total of r^2 two-dimensional scaling functions are obtained:

$$\Phi^{(s,t)}(x,y) := \phi^s(x)\phi^t(y) \tag{3.158}$$

where $s,t \in \{1,\ldots,r\}$. Given r associated wavelets $\{\psi^1,\ldots,\psi^r\}$, three different types of tensor product wavelets are obtained, taking the form

$$\Psi^{1,(s,t)}(x,y) := \phi^s(x)\psi^t(y) \tag{3.159}$$

$$\Psi^{2,(s,t)}(x,y) := \psi^s(x)\phi^t(y) \tag{3.160}$$

$$\Psi^{3,(s,t)}(x,y) := \psi^s(x)\psi^t(y) \tag{3.161}$$

where, once again, $s,t \in \{1,\ldots,r\}$. Therefore, there are a total of $3r^2$ two-dimensional wavelets. It is convenient to introduce the notation:

$$\begin{aligned}
\Phi^{(s,t)}_{j,(k,m)}(x,y) &:= \phi^s_{j,k}(x)\phi^t_{j,m}(y) \\
\Psi^{1,(s,t)}_{j,(k,m)}(x,y) &:= \phi^s_{j,k}(x)\psi^t_{j,m}(y) \\
\Psi^{2,(s,t)}_{j,(k,m)}(x,y) &:= \psi^s_{j,k}(x)\phi^t_{j,m}(y) \\
\Psi^{3,(s,t)}_{j,(k,m)}(x,y) &:= \psi^s_{j,k}(x)\psi^t_{j,m}(y)
\end{aligned} \tag{3.162}$$

where, as before,

$$\phi^s_{j,k}(x) = 2^{j/2}\phi^s\left(2^j x - k\right) \tag{3.163}$$

$$\psi^s_{j,k}(x) = 2^{j/2}\psi^s\left(2^j x - k\right) \tag{3.164}$$

Note, once again, that there are three types of wavelets. The superscript (1,2, or 3) indicates whether the wavelet is formed as the tensor product of a scaling function in the first dimension and a wavelet in the second dimension (type 1), a wavelet in the first dimension and a scaling function in the second dimension (type 2), or a wavelet

in both dimensions (type 3). In addition, each of the functions in equation (3.162) has a superscript (s,t) which indicates the individual scaling functions or wavelets that compose each two-dimensional function. Just as in the one-dimensional case, the subscript j represents the discretization level. In two dimensions, we have a grid composed of 2^j x 2^j square elements. Finally, the two-dimensional scaling functions and wavelets are indexed by a translate (k,m). The index k indicates the position of the function in the first dimension while the index m gives the position of the function in the second dimension.

Tensor product wavelets are amenable to multiresolution analysis in much the same manner as their one-dimensional counterparts. We define the two-dimensional spaces

$$V_j := span\left\{\Phi^{(s,t)}_{j,(k,m)} : \ s,t \in \{1,\ldots,r\}\right\}_{k,m\in\mathbb{Z}} \tag{3.165}$$

$$W^1_j := span\left\{\Psi^{1,(s,t)}_{j,(k,m)} : \ s,t \in \{1,\ldots,r\}\right\}_{k,m\in\mathbb{Z}} \tag{3.166}$$

$$W^2_j := span\left\{\Psi^{2,(s,t)}_{j,(k,m)} : \ s,t \in \{1,\ldots,r\}\right\}_{k,m\in\mathbb{Z}} \tag{3.167}$$

$$W^3_j := span\left\{\Psi^{3,(s,t)}_{j,(k,m)} : \ s,t \in \{1,\ldots,r\}\right\}_{k,m\in\mathbb{Z}} \tag{3.168}$$

Then, the space V_j can be decomposed as follows:

$$V_j = V_{j-1} \oplus W^1_{j-1} \oplus W^2_{j-1} \oplus W^3_{j-1} \tag{3.169}$$

If the decomposition is carried out over several levels, we have

$$V_j = V_{j_0} \bigoplus_{l=j_0}^{j-1} \{W^1_l \oplus W^2_l \oplus W^3_l\} \tag{3.170}$$

where j_0 is the coarsest level used in the multiscale expansion.

A single-scale representation of a two-dimensional function f in terms of the basis functions of V_j can be written as

$$f_j(x,y) = \sum_{k,m,s,t} \alpha^{(s,t)}_{j,(k,m)} \Phi^{(s,t)}_{j,(k,m)}(x,y) \tag{3.171}$$

where $\{\alpha^{(s,t)}_{j,(k,m)}\}$ are constant scaling function coefficients. Equation (3.169) enables us to write an equivalent two-scale expansion of the form

$$\begin{aligned} f_j(x,y) &= \sum_{k,m,s,t} \alpha^{(s,t)}_{j-1,(k,m)} \Phi^{(s,t)}_{j-1,(k,m)}(x,y) + \sum_{k,m,s,t} \beta^{1,(s,t)}_{j-1,(k,m)} \Psi^{1,(s,t)}_{j-1,(k,m)}(x,y) \\ &\quad + \sum_{k,m,s,t} \beta^{2,(s,t)}_{j-1,(k,m)} \Psi^{2,(s,t)}_{j-1,(k,m)}(x,y) + \sum_{k,m,s,t} \beta^{3,(s,t)}_{j-1,(k,m)} \Psi^{3,(s,t)}_{j-1,(k,m)}(x,y) \end{aligned} \tag{3.172}$$

where $\{\beta^{r,(s,t)}_{j,(k,m)}\}$ are constant wavelet coefficients. A multiscale expansion of f_j, corresponding to equation (3.170), can be written as

$$\begin{aligned} f_j(x,y) &= \sum_{k,m,s,t} \alpha^{(s,t)}_{j_0,(k,m)} \Phi^{(s,t)}_{j_0,(k,m)}(x,y) + \sum_{l=j_0}^{j-1} \sum_{k,m,s,t} \beta^{1,(s,t)}_{l,(k,m)} \Psi^{1,(s,t)}_{l,(k,m)}(x,y) \\ &\quad + \sum_{l=j_0}^{j-1} \sum_{k,m,s,t} \beta^{2,(s,t)}_{l,(k,m)} \Psi^{2,(s,t)}_{l,(k,m)}(x,y) + \sum_{l=j_0}^{j-1} \sum_{k,m,s,t} \beta^{3,(s,t)}_{l,(k,m)} \Psi^{3,(s,t)}_{l,(k,m)}(x,y) \end{aligned} \tag{3.173}$$

Two-dimensional decomposition formulas can be derived in a similar manner as the one-dimensional case. A decomposition formula for the coefficients $\{\alpha^{(s,t)}_{j-1,(k,m)}\}$ in equation (3.172) can be obtained by multiplying the single-scale and two-scale representations of f_j by $\Phi^{(u,v)}_{j-1,(a,c)}$ and integrating over the two-dimensional space $\mathbb{R}^2$:

$$\begin{aligned} &\int_{\mathbb{R}^2} \sum_{k,m,s,t} \alpha^{(s,t)}_{j,(k,m)} \Phi^{(s,t)}_{j,(k,m)}(x,y) \Phi^{(u,v)}_{j-1,(a,c)}(x,y) dxdy \\ &\qquad = \int_{\mathbb{R}^2} \sum_{k,m,s,t} \alpha^{(s,t)}_{j-1,(k,m)} \Phi^{(s,t)}_{j-1,(k,m)}(x,y) \Phi^{(u,v)}_{j-1,(a,c)}(x,y) dxdy \end{aligned}$$

$$+\int_{\mathbb{R}^2}\sum_{k,m,s,t}\beta^{1,(s,t)}_{j-1,(k,m)}\Psi^{1,(s,t)}_{j-1,(k,m)}(x,y)\Phi^{(u,v)}_{j-1,(a,c)}(x,y)dxdy$$
$$+\int_{\mathbb{R}^2}\sum_{k,m,s,t}\beta^{2,(s,t)}_{j-1,(k,m)}\Psi^{2,(s,t)}_{j-1,(k,m)}(x,y)\Phi^{(u,v)}_{j-1,(a,c)}(x,y)dxdy$$
$$+\int_{\mathbb{R}^2}\sum_{k,m,s,t}\beta^{3,(s,t)}_{j-1,(k,m)}\Psi^{3,(s,t)}_{j-1,(k,m)}(x,y)\Phi^{(u,v)}_{j-1,(a,c)}(x,y)dxdy$$

Using the expanded form of the functions from equations (3.158) through (3.161) and rearranging the integrals, we have

$$\sum_{k,m,s,t}\alpha^{(s,t)}_{j,(k,m)}\int_{\mathbb{R}}\phi^s_{j,k}(x)\phi^u_{j-1,a}(x)dx\int_{\mathbb{R}}\phi^t_{j,m}(y)\phi^v_{j-1,c}(y)dy \tag{3.174}$$
$$=\sum_{k,m,s,t}\alpha^{(s,t)}_{j-1,(k,m)}\int_{\mathbb{R}}\phi^s_{j-1,k}(x)\phi^u_{j-1,a}(x)dx\int_{\mathbb{R}}\phi^t_{j-1,m}(y)\phi^v_{j-1,c}(y)dy$$
$$+\sum_{k,m,s,t}\beta^{1,(s,t)}_{j-1,(k,m)}\int_{\mathbb{R}}\phi^s_{j-1,k}(x)\phi^u_{j-1,a}(x)dx\int_{\mathbb{R}}\psi^t_{j-1,m}(y)\phi^v_{j-1,c}(y)dy$$
$$+\sum_{k,m,s,t}\beta^{2,(s,t)}_{j-1,(k,m)}\int_{\mathbb{R}}\psi^s_{j-1,k}(x)\phi^u_{j-1,a}(x)dx\int_{\mathbb{R}}\phi^t_{j-1,m}(y)\phi^v_{j-1,c}(y)dy$$
$$+\sum_{k,m,s,t}\beta^{3,(s,t)}_{j-1,(k,m)}\int_{\mathbb{R}}\psi^s_{j-1,k}(x)\phi^u_{j-1,a}(x)dx\int_{\mathbb{R}}\psi^t_{j-1,m}(y)\phi^v_{j-1,c}(y)dy$$

Recall that multiwavelets satisfy the following orthogonality properties in one dimension:

$$(1)\qquad \int_{\mathbb{R}}\phi^s_{j,k}(x)\phi^t_{j,m}(x)dx=\delta_{(k,s),(m,t)}$$
$$(2)\qquad \int_{\mathbb{R}}\psi^s_{j,k}(x)\psi^t_{j,m}(x)dx=\delta_{(k,s),(m,t)} \tag{3.175}$$
$$(3)\qquad \int_{\mathbb{R}}\phi^s_{j,k}(x)\psi^t_{j,m}(x)dx=0\quad\forall\ k,m\in\mathbb{Z},\quad s,t\in\{1,\ldots,r\},\quad j\le l$$

Using these properties, equation (3.174) can be reduced to

$$\sum_{k,m,s,t} \alpha^{(s,t)}_{j-1,(k,m)} \delta_{(k,s),(a,u)} \delta_{(m,t),(c,v)}$$
$$= \sum_{k,m,s,t} \alpha^{(s,t)}_{j,(k,m)} \int_{\mathbb{R}} \phi^s_{j,k}(x) \phi^u_{j-1,a}(x) dx \int_{\mathbb{R}} \phi^t_{j,m}(y) \phi^v_{j-1,c}(y) dy$$

and

$$\alpha^{(u,v)}_{j-1,(a,c)} = \sum_{k,m,s,t} \alpha^{(s,t)}_{j,(k,m)} \int_{\mathbb{R}} \phi^s_{j,k}(x) \phi^u_{j-1,a}(x) dx \int_{\mathbb{R}} \phi^t_{j,m}(y) \phi^v_{j-1,c}(y) dy \tag{3.176}$$

Recall that we have the following two-scale equations for multiwavelets:

$$\phi^s_{j-1,k}(x) = \sum_{p,q} a^{s,q}_p \phi^q_{j,2k+p}(x) \tag{3.177}$$
$$\psi^s_{j-1,k}(x) = \sum_{p,q} b^{s,q}_p \phi^q_{j,2k+p}(x) \tag{3.178}$$

Then, using equation (3.177) to substitute for $\phi^u_{j-1,a}$ and $\phi^v_{j-1,c}$, equation (3.176) becomes

$$\alpha^{(u,v)}_{j-1,(a,c)} = \sum_{k,m,s,t} \sum_{p,q,f,e} \alpha^{(s,t)}_{j,(k,m)} a^{u,q}_p a^{v,e}_f \int_{\mathbb{R}} \phi^s_{j,k}(x) \phi^q_{j,2a+p}(x) dx \int_{\mathbb{R}} \phi^t_{j,m}(y) \phi^e_{j,2c+f}(y) dy$$

Using the orthogonality properties once again, this can be written as

$$\alpha^{(u,v)}_{j-1,(a,c)} = \sum_{k,m,s,t} \sum_{p,q,f,e} \alpha^{(s,t)}_{j,(k,m)} a^{u,q}_p a^{v,e}_f \delta_{(2a+p,q),(k,s)} \delta_{(2c+f,e),(m,t)}$$

Finally, we obtain the following decomposition formula for the scaling function coefficients on level $j-1$:

$$\alpha^{(u,v)}_{j-1,(a,c)} = \sum_{k,m,s,t} a^{u,s}_{k-2a} a^{v,t}_{m-2c} \alpha^{(s,t)}_{j,(k,m)} \tag{3.179}$$

In a similar manner, the other two-dimensional decomposition formulas can be derived:

$$\beta^{1,(u,v)}_{j-1,(a,c)} = \sum_{k,m,s,t} a^{u,s}_{k-2a} b^{v,t}_{m-2c} \alpha^{(s,t)}_{j,(k,m)} \tag{3.180}$$

$$\beta^{2,(u,v)}_{j-1,(a,c)} = \sum_{k,m,s,t} b^{u,s}_{k-2a} a^{v,t}_{m-2c} \alpha^{(s,t)}_{j,(k,m)} \tag{3.181}$$

$$\beta^{3,(u,v)}_{j-1,(a,c)} = \sum_{k,m,s,t} b^{u,s}_{k-2a} b^{v,t}_{m-2c} \alpha^{(s,t)}_{j,(k,m)} \tag{3.182}$$

These formulas are obtained by multiplying equations (3.171) and (3.172) by $\Psi^{1,(u,v)}_{j-1,(a,c)}$, $\Psi^{2,(u,v)}_{j-1,(a,c)}$, and $\Psi^{3,(u,v)}_{j-1,(a,c)}$, respectively, and integrating over $\mathbb{R}^2$. Once again, the orthogonality properties of the multiwavelets and the two-scale equations are used to simplify the integrations. The above decomposition formulas can be applied recursively to calculate the coefficients that appear in the multiscale expansion of f_j in equation (3.170).

A reconstruction formula for the single-scale coefficients $\{\alpha^{(s,t)}_{j,(k,m)}\}$ can be derived by multiplying the single-scale and two-scale expansions of f_j by $\Phi^{(u,v)}_{j,(a,c)}$ and integrating over $\mathbb{R}^2$:

$$\begin{aligned}
&\sum_{k,m,s,t} \alpha^{(s,t)}_{j,(k,m)} \int_{\mathbb{R}} \phi^s_{j,k}(x)\phi^u_{j,a}(x)dx \int_{\mathbb{R}} \phi^t_{j,m}(y)\phi^v_{j,c}(y)dy \qquad (3.183)\\
&\quad = \sum_{k,m,s,t} \alpha^{(s,t)}_{j-1,(k,m)} \int_{\mathbb{R}} \phi^s_{j-1,k}(x)\phi^u_{j,a}(x)dx \int_{\mathbb{R}} \phi^t_{j-1,m}(y)\phi^v_{j,c}(y)dy\\
&\quad + \sum_{k,m,s,t} \beta^{1,(s,t)}_{j-1,(k,m)} \int_{\mathbb{R}} \phi^s_{j-1,k}(x)\phi^u_{j,a}(x)dx \int_{\mathbb{R}} \psi^t_{j-1,m}(y)\phi^v_{j,c}(y)dy\\
&\quad + \sum_{k,m,s,t} \beta^{2,(s,t)}_{j-1,(k,m)} \int_{\mathbb{R}} \psi^s_{j-1,k}(x)\phi^u_{j,a}(x)dx \int_{\mathbb{R}} \phi^t_{j-1,m}(y)\phi^v_{j,c}(y)dy\\
&\quad + \sum_{k,m,s,t} \beta^{3,(s,t)}_{j-1,(k,m)} \int_{\mathbb{R}} \psi^s_{j-1,k}(x)\phi^u_{j,a}(x)dx \int_{\mathbb{R}} \psi^t_{j-1,m}(y)\phi^v_{j,c}(y)dy
\end{aligned}$$

Note that the expanded form of the scaling functions and wavelets has been used in equation (3.183). By the orthogonality properties of the multiwavelets, equation (3.183) can be simplified to

$$\begin{aligned}
\alpha_{j,(a,c)}^{(u,v)} = & \sum_{k,m,s,t} \alpha_{j-1,(k,m)}^{(s,t)} \int_{\mathbb{R}} \phi_{j-1,k}^{s}(x)\phi_{j,a}^{u}(x)dx \int_{\mathbb{R}} \phi_{j-1,m}^{t}(y)\phi_{j,c}^{v}(y)dy \\
& + \sum_{k,m,s,t} \beta_{j-1,(k,m)}^{1,(s,t)} \int_{\mathbb{R}} \phi_{j-1,k}^{s}(x)\phi_{j,a}^{u}(x)dx \int_{\mathbb{R}} \psi_{j-1,m}^{t}(y)\phi_{j,c}^{v}(y)dy \\
& + \sum_{k,m,s,t} \beta_{j-1,(k,m)}^{2,(s,t)} \int_{\mathbb{R}} \psi_{j-1,k}^{s}(x)\phi_{j,a}^{u}(x)dx \int_{\mathbb{R}} \phi_{j-1,m}^{t}(y)\phi_{j,c}^{v}(y)dy \\
& + \sum_{k,m,s,t} \beta_{j-1,(k,m)}^{3,(s,t)} \int_{\mathbb{R}} \psi_{j-1,k}^{s}(x)\phi_{j,a}^{u}(x)dx \int_{\mathbb{R}} \psi_{j-1,m}^{t}(y)\phi_{j,c}^{v}(y)dy
\end{aligned}$$

Using the two-scale relations, we have

$$\begin{aligned}
\alpha_{j,(a,c)}^{(u,v)} = & \sum_{k,m,s,t} \sum_{p,q,f,e} a_p^{s,q} a_f^{t,e} \alpha_{j-1,(k,m)}^{(s,t)} \int_{\mathbb{R}} \phi_{j,2k+p}^{q}(x)\phi_{j,a}^{u}(x)dx \int_{\mathbb{R}} \phi_{j,2m+f}^{e}(y)\phi_{j,c}^{v}(y)dy \\
& + \sum_{k,m,s,t} \sum_{p,q,f,e} a_p^{s,q} b_f^{t,e} \beta_{j-1,(k,m)}^{1,(s,t)} \int_{\mathbb{R}} \phi_{j,2k+p}^{q}(x)\phi_{j,a}^{u}(x)dx \int_{\mathbb{R}} \phi_{j,2m+f}^{e}(y)\phi_{j,c}^{v}(y)dy \\
& + \sum_{k,m,s,t} \sum_{p,q,f,e} b_p^{s,q} a_f^{t,e} \beta_{j-1,(k,m)}^{2,(s,t)} \int_{\mathbb{R}} \phi_{j,2k+p}^{q}(x)\phi_{j,a}^{u}(x)dx \int_{\mathbb{R}} \phi_{j,2m+f}^{e}(y)\phi_{j,c}^{v}(y)dy \\
& + \sum_{k,m,s,t} \sum_{p,q,f,e} b_p^{s,q} b_f^{t,e} \beta_{j-1,(k,m)}^{3,(s,t)} \int_{\mathbb{R}} \phi_{j,2k+p}^{q}(x)\phi_{j,a}^{u}(x)dx \int_{\mathbb{R}} \phi_{j,2m+f}^{e}(y)\phi_{j,c}^{v}(y)dy
\end{aligned}$$

Once again making use of the orthogonality properties, this reduces to

$$\begin{aligned}
\alpha_{j,(a,c)}^{(u,v)} = & \sum_{k,m,s,t} \sum_{p,q,f,e} a_p^{s,q} a_f^{t,e} \alpha_{j-1,(k,m)}^{(s,t)} \delta_{(2k+p,q),(a,u)} \delta_{(2m+f,e),(c,v)} \\
& + \sum_{k,m,s,t} \sum_{p,q,f,e} a_p^{s,q} b_f^{t,e} \beta_{j-1,(k,m)}^{1,(s,t)} \delta_{(2k+p,q),(a,u)} \delta_{(2m+f,e),(c,v)} \\
& + \sum_{k,m,s,t} \sum_{p,q,f,e} b_p^{s,q} a_f^{t,e} \beta_{j-1,(k,m)}^{2,(s,t)} \delta_{(2k+p,q),(a,u)} \delta_{(2m+f,e),(c,v)}
\end{aligned}$$

$$+\sum_{k,m,s,t}\sum_{p,q,f,e} b_p^{s,q} b_f^{t,e}\, \beta_{j-1,(k,m)}^{3,(s,t)} \delta_{(2k+p,q),(a,u)} \delta_{(2m+f,e),(c,v)}$$

Finally, the following reconstruction formula is obtained:

$$\alpha_{j,(a,c)}^{(u,v)} = \sum_{k,m,s,t} \Big\{ a_{a-2k}^{s,u} a_{c-2m}^{t,v} \alpha_{j-1,(k,m)}^{(s,t)} + a_{a-2k}^{s,u} b_{c-2m}^{t,v} \beta_{j-1,(k,m)}^{1,(s,t)} + b_{a-2k}^{s,u} a_{c-2m}^{t,v} \beta_{j-1,(k,m)}^{2,(s,t)} + b_{a-2k}^{s,u} b_{c-2m}^{t,v} \beta_{j-1,(k,m)}^{3,(s,t)} \Big\} \tag{3.184}$$

Single-generator wavelet families, which are comprised of one scaling function ϕ and one associated wavelet ψ, clearly form a subset of multiwavelets. In this case, we have one two-dimensional tensor product scaling function and three wavelets:

$$\begin{aligned} \Phi(x,y) &:= \phi(x)\phi(y) \\ \Psi^1(x,y) &:= \phi(x)\psi(y) \\ \Psi^2(x,y) &:= \psi(x)\phi(y) \\ \Psi^3(x,y) &:= \psi(x)\psi(y) \end{aligned} \tag{3.185}$$

A single-scale approximation of a two-dimensional function f can be written as

$$f_j(x,y) = \sum_{k,m} \alpha_{j,(k,m)} \Phi_{j,(k,m)}(x,y) + \sum_{k,m} \beta_{j,(k,m)}^1 \Psi_{j,(k,m)}^1(x,y) + \sum_{k,m} \beta_{j,(k,m)}^2 \Psi_{j,(k,m)}^2(x,y) + \sum_{k,m} \beta_{j,(k,m)}^3 \Psi_{j,(k,m)}^3(x,y) \tag{3.186}$$

and an equivalent, multiscale representation is given by

$$f_j(x,y) = \sum_{k,m} \alpha_{j_0,(k,m)} \Phi_{j_0,(k,m)}(x,y) + \sum_{l=j_0}^{j-1} \sum_{k,m} \beta_{l,(k,m)}^1 \Psi_{l,(k,m)}^1(x,y) \tag{3.187}$$

$$+\sum_{l=j_0}^{j-1}\sum_{k,m}\beta^2_{l,(k,m)}\Psi^2_{l,(k,m)}(x,y)+\sum_{l=j_0}^{j-1}\sum_{k,m}\beta^3_{l,(k,m)}\Psi^3_{l,(k,m)}(x,y)$$

In this case, the decomposition formulas take the form:

$$\alpha_{j-1,(a,c)} = \sum_{k,m} a_{k-2a}a_{m-2c}\alpha_{j,(k,m)} \tag{3.188}$$

$$\beta^1_{j-1,(a,c)} = \sum_{k,m} a_{k-2a}b_{m-2c}\alpha_{j,(k,m)} \tag{3.189}$$

$$\beta^2_{j-1,(a,c)} = \sum_{k,m} b_{k-2a}a_{m-2c}\alpha_{j,(k,m)} \tag{3.190}$$

$$\beta^3_{j-1,(a,c)} = \sum_{k,m} b_{k-2a}b_{m-2c}\alpha_{j,(k,m)} \tag{3.191}$$

and the reconstruction formula can be written as

$$\begin{aligned}\alpha_{j,(a,c)} = \sum_{k,m} \{&a_{a-2k}a_{c-2m}\alpha_{j-1,(k,m)} + a_{a-2k}b_{c-2m}\beta^1_{j-1,(k,m)} \\ &+b_{a-2k}a_{c-2m}\beta^2_{j-1,(k,m)} + b_{a-2k}b_{c-2m}\beta^3_{j-1,(k,m)}\}\end{aligned} \tag{3.192}$$

Recall that, while in the general multiwavelet case, the scaling function and wavelet filters are matrices, in the single-generator case they reduce to scalars.

As a final note, it is not difficult to extend the results given in this section to higher-dimensional tensor product multiwavelets. The notation necessarily becomes more cumbersome, but the concept remains the same. In general, for a multiwavelet family generated by r scaling functions, there are a total of r^n n-dimensional tensor product scaling functions and $3r^n$ associated wavelets.

CHAPTER 4
MULTIWAVELET CONSTRUCTIONS

In this chapter, piecewise-polynomial multiwavelets are constructed from classical finite elements using the technique of intertwining multiresolution analyses. These multiwavelets are orthonormal, compactly supported, and possess symmetry or antisymmetry. Furthermore, they can be constructed to have arbitrary approximation order and are easily adapted to boundaries. In two dimensions, the resulting tensor product multiwavelets are easily adapted to the square domain over which the symmetric form of the second-order Volterra kernel is supported. In Section 4.1, the technique of intertwining multiresolution analyses is introduced. It is shown that one intertwining is not sufficient to generate multiwavelets with desirable approximation properties from lower-order finite elements. In Section 4.2, it is demonstrated that, as noted by Donovan, Geronimo, and Hardin [2], by performing two successive intertwinings of multiresolution analyses, multiwavelets that are suitable for approximation can be derived from these finite elements. This technique is demonstrated in the construction of piecewise-quadratic and piecewise-cubic multiwavelets in Sections 4.3 and 4.4, respectively. In Section 4.5, a theorem is given that generalizes these results for the construction of piecewise-polynomial wavelets of arbitrary approximation order. In order to use these piecewise-polynomial multiwavelets to represent first and second-order Volterra kernels, it is necessary to adapt the functions to the domains over which the kernels are supported. Therefore, boundary-adapted multiwavelets are discussed in Section 4.6. Finally, in Section 4.7, an example is given in which the boundary-adapted multiwavelets are used to decompose a one-dimensional function.

4.1 Multiwavelets and Multiresolution

Recall, from Chapter 3, that a multiresolution analysis of multiplicity r is defined as a family of subspaces $\{V_j\}_{j\in\mathbb{Z}}$ of $L^2(\mathbb{R})$ satisfying the following conditions:

$$
\begin{aligned}
&(1) \quad \cdots V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \cdots \\
&(2) \quad \bigcap_{j\in\mathbb{Z}} V_j = \{0\} \\
&(3) \quad \overline{\bigcup_{j\in\mathbb{Z}} V_j} = L^2(\mathbb{R}) \\
&(4) \quad f(x) \in V_j \Longleftrightarrow f(2x) \in V_{j+1} \\
&(5) \quad f(x) \in V_j \Longrightarrow f(x-k) \in V_j \quad \forall\, k \in \mathbb{Z} \\
&(6) \quad \text{There are } r \text{ functions } \{\phi^1, \ldots, \phi^r\} \text{ such that the set of translates} \\
&\qquad \{\phi^s(x-k) : \ s = 1, \ldots, r\}_{k\in\mathbb{Z}} \text{ forms a Riesz basis for } V_0
\end{aligned}
\tag{4.1}
$$

The functions $\{\phi^1, \ldots, \phi^r\}$ are the scaling functions, or generators, of the multiresolution analysis. Each of the spaces $\{V_j\}$ above is an example of a finitely-generated shift-invariant (FSI) space. As noted in Ref. [2], there is clearly some flexibility in the selection of the space V_0 in any given multiresolution analysis. In this development, the generators of V_0 are chosen to have support in $[-1, 1]$. This simply amounts to shifting the multiresolution sequence $\{V_j\}$ so that we consider $\{\tilde{V}_s\}_{s\in\mathbb{Z}}$, where $\tilde{V}_s = V_{j+m}$ for some fixed $m \in \mathbb{Z}$. In addition, it will be required that the families of generators $\{\phi^1, \ldots, \phi^r\}$ are minimally supported in $[-1, 1]$.

Definition 4.1 [2]: Suppose that $\{\phi^1, \ldots, \phi^r\}$ are generators of a multiresolution analysis, where $\{\phi^1, \ldots, \phi^k\}$ are supported on $[-1, 1]$ and the remaining $\{\phi^{k+1}, \ldots, \phi^r\}$ are supported on $[0, 1]$. The generators $\{\phi^1, \ldots, \phi^r\}$ are *minimally supported* on $[-1, 1]$ provided that

(1) $\{\phi^1, \ldots, \phi^k\}$ are linearly independent on $[0,1]$

(2) $\{\phi^1, \ldots, \phi^k\}$ are linearly independent on $[-1,0]$

(3) $\{0\} = span\left(\{\phi^1(x)\chi_{[0,1]}(x), \ldots, \phi^k(x)\chi_{[0,1]}(x)\} \bigcap \{\phi^1(x)\chi_{[0,1]}(x-1), \ldots, \phi^k(x)\chi_{[0,1]}(x-1)\}\right)$

Note that $\chi_{[0,1]}$ denotes the characteristic function over $[0,1]$, defined as

$$\chi_{[0,1]}(x) := \begin{cases} 1 & x \in [0,1] \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

The advantage of working with minimally-supported families of generators is that the restrictions of shifts of the generators $\{\phi^1, \ldots, \phi^r\}$ to any interval are linearly independent. That is,

$$\phi^1, \ldots, \phi^k, \phi^1(\cdot - 1), \ldots, \phi^k(\cdot - 1), \phi^{k+1}, \ldots, \phi^r$$

are all linearly independent on $[0,1]$.

The fundamental tool that will be used in the construction of orthogonal piecewise-polynomial multiwavelets is the technique of intertwining multiresolution analyses. Suppose that $\{\phi^1, \ldots, \phi^r\}$ are generators of a multiresolution analysis that is minimally supported on $[-1,1]$. We define the spaces

$$A(V_0) := span\{\phi^s : s = k+1, \ldots, r\} \tag{4.3}$$

$$B_\sigma(V_0) := span\left(\{\phi^s(\cdot - \sigma)\chi_{[0,1]} : s = 1, \ldots, k\} \bigcup A(V_0)\right), \quad \sigma = 0, 1 \tag{4.4}$$

$$C_\sigma(V_0) := B_\sigma(V_0) \ominus A(V_0), \quad \sigma = 0, 1 \tag{4.5}$$

The process by which intertwining multiresolution analyses are constructed is defined in terms of the subspaces $A(V_0)$, $B_\sigma(V_0)$, and $C_\sigma(V_0)$ defined above and the following lemma from Ref. [2]:

Lemma 4.1 (Donovan, Geronimo, and Hardin [2]): Let $\{V_j\}_{j\in\mathbb{Z}}$ be a multiresolution analysis generated by r scaling functions $\{\phi^1,\ldots,\phi^r\}$ that are minimally supported on $[-1,1]$. Suppose that there is a subspace $W \subset A(V_1)\ominus A(V_0)$ such that

$$(I-P_W)\,C_0(V_0)\perp(I-P_W)\,C_1(V_0) \tag{4.6}$$

where P_W is the orthogonal projection onto W. Suppose that $\{w_1,\ldots,w_k\}$ is a basis for W. Then, the family of functions $\{\phi^1,\ldots,\phi^r,w_1,\ldots,w_k\}$ generates an orthogonal multiresolution analysis $\{\tilde{V}_j\}$ that intertwines $\{V_j\}$ in the sense that

$$\cdots V_0\subset\tilde{V}_0\subset V_1\cdots$$

The method by which this theorem can be utilized to generate orthogonal multiwavelets is simple, at least in principle. We begin by considering the multiresolution analysis $\{V_j\}$ that is generated by classical C_0 finite elements supported on $[-1,1]$. We then seek a basis $\{w_1,\ldots,w_k\}$ for $A(V_1)\ominus A(V_0)$ that satisfies equation (4.6). If such a basis can be found, the desired orthogonal multiresolution analysis has been obtained. This construction is depicted schematically in Figure (4.1). Unfortunately, in many cases of interest, the subspace $A(V_1)\ominus A(V_0)$ is not large enough to design multiwavelets that exhibit desirable properties such as symmetry or antisymmetry. For example, the construction carried out in Ref. [2] yields smooth piecewise-polynomial multiwavelets that are of cubic order or higher. The lack of lower-order orthogonal piecewise-polynomial multiwavelets can be attributed precisely to the small dimension of $A(V_1)\ominus A(V_0)$.

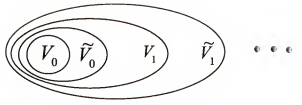


Figure 4.1: Intertwining Multiresolution Analyses

To explain the role of the dimension of $A(V_1) \ominus A(V_0)$ in the construction of multiwavelets, consider the two examples depicted in Figures (4.2) and (4.3). Figure (4.2) depicts the spaces $A(V_0)$, $A(V_1)$, and $A(V_2)$ for the case in which we seek to construct intertwining multiresolution analyses from piecewise-linear finite elements. Figure (4.3) depicts the corresponding spaces for classical quadratic finite elements. In either case, the dimension of the space $A(V_1) \ominus A(V_0)$ is 2. If we seek a subspace $W \subseteq A(V_1) \ominus A(V_0)$ that satisfies equation (4.6), the dimension of W is 1. In other words, the subspace W is completely determined by the choice of the original nested spaces of piecewise polynomials in Figures (4.2) and (4.3), and the orthogonality condition in equation (4.6). As a result, there is no freedom to design the resulting basis for W to be reasonable for approximation purposes. In fact, Example (3.1) in Ref. [2] carries out this analysis starting with piecewise-linear finite elements. The scaling functions and wavelets associated with the resulting orthogonal multiresolution analysis are neither symmetric nor antisymmetric. Moreover, they do not exhibit appropriate interpolatory properties at the element boundaries and are difficult to employ for general classes of boundary conditions. They simply are not attractive candidates for approximation.

4.2 Construction of Intertwining Multiresolution Analyses

From the preceding discussion, it is apparent that the construction of intertwining multiresolution analyses from lower-order finite elements does not yield functions with good approximation properties. As noted in Refs. [2] and [3], this problem can be addressed by creating two successive intertwinings of multiresolution analyses. The

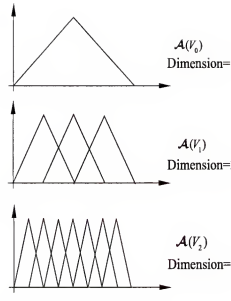


Figure 4.2: Generators Supported on [0,1]: Linear FEM

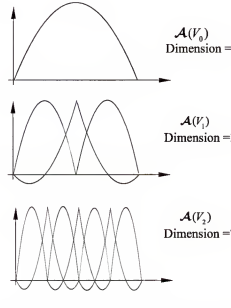


Figure 4.3: Generators Supported on [0,1]: Quadratic FEM

second intertwining effectively gives another degree of freedom to the design process. In this section, this procedure is outlined for the goal of constructing piecewise-polynomial multiwavelets from the classical finite element spaces.

The starting point for the procedure is the finite shift-invariant (FSI) spaces $\{V_j\}_{j\in\mathbb{Z}}$ associated with the classical finite element spaces of a particular order. Given r generators $\{\phi^1,\ldots,\phi^r\}$, the FSI spaces are defined as

$$V_j := span\left\{\phi^s_{j,k}: \quad s=1,\ldots,r\right\}_{k\in\mathbb{Z}} \tag{4.7}$$

where we have the usual notation

$$\phi^s_{j,k}(x) = 2^{j/2}\phi^s\left(2^j x - k\right) \tag{4.8}$$

The first step in the intertwining process is to select a function $w \in V_1$. Then, a new set of generators is defined as

$$\left\{\tilde{\phi}^1,\ldots,\tilde{\phi}^{r+1}\right\} := \left\{\phi^1,\ldots,\phi^r,w\right\} \tag{4.9}$$

for a new multiresolution analysis $\{\tilde{V}_j\}_{j\in\mathbb{Z}}$. This multiresolution analysis satisfies

$$V_0 \subset \tilde{V}_0 \subset V_1 \tag{4.10}$$

By the definition of a finite shift-invariant space, we also have

$$V_0 \subset \tilde{V}_0 \subset V_1 \subset \tilde{V}_1 \cdots \tag{4.11}$$

and $\{\tilde{V}_j\}$ is a multiresolution analysis that intertwines $\{V_j\}$. Note $\{\tilde{V}_j\}$ is not constructed such that it is an orthogonal multiresolution analysis. If that were the case, the dimensionality of $A(V_1) \ominus A(V_0)$ would severely restrict the design of w, and hence of the multiresolution analysis $\{\tilde{V}_j\}$.

After the construction of $\{\tilde{V}_p\}$, a second intertwining is performed. That is, another function $\tilde{w} \in \tilde{V}_1$ is chosen and a new set of generators is defined:

$$\left\{\tilde{\tilde{\phi}}^1, \ldots, \tilde{\tilde{\phi}}^{r+2}\right\} := \left\{\tilde{\phi}^1, \ldots, \tilde{\phi}^{r+1}, \tilde{w}\right\} \tag{4.12}$$

In this case, $\tilde{w}$ is chosen to satisfy the conditions of Lemma (4.1). In other words, we seek $\tilde{w} \in A(\tilde{V}_1) \ominus A(\tilde{V}_0)$ such that

$$(I - P_{\tilde{W}})\, C_0(\tilde{V}_0) \perp (I - P_{\tilde{W}})\, C_1(\tilde{V}_0) \tag{4.13}$$

With the second intertwining, we have

$$\tilde{V}_0 \subset \tilde{\tilde{V}}_0 \subset \tilde{V}_1 \tag{4.14}$$

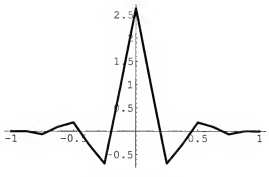
and $\{\tilde{\tilde{V}}_j\}$ is an orthogonal multiresolution analysis. In terms of the original multiresolution $\{V_j\}$, we have

$$\begin{aligned} V_0 &\subset \tilde{V}_0 \subset \tilde{\tilde{V}}_0 \\ V_1 &\subset \tilde{V}_1 \subset \tilde{\tilde{V}}_1 \\ V_2 &\subset \tilde{V}_2 \subset \tilde{\tilde{V}}_2 \\ &\vdots \end{aligned} \tag{4.15}$$

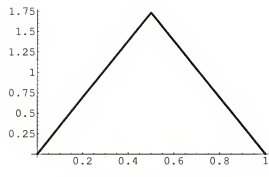
and

$$\tilde{\tilde{V}}_0 \subset \tilde{\tilde{V}}_1 \subset \tilde{\tilde{V}}_2 \tag{4.16}$$

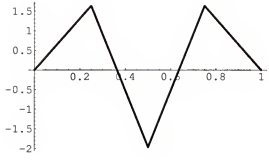
Therefore, by performing two intertwinings of multiresolution analyses, an orthogonal multiresolution analysis can be obtained. Donovan, Geronimo, and Hardin [2] have demonstrated this procedure in deriving piecewise-linear multiwavelets from the classical linear finite element spaces. The derived piecewise-linear scaling functions and



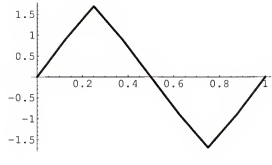
Scaling Function ϕ^1



Scaling Function ϕ^2



Scaling Function ϕ^3



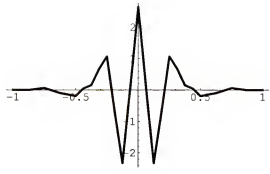
Scaling Function ϕ^4

Figure 4.4: Piecewise-Linear Scaling Functions [2]

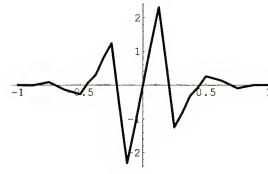
wavelets are depicted in Figures (4.4) and (4.5). The corresponding scaling function and wavelet filter matrices are listed in Tables (4.1) and (4.2), respectively. In the following sections, the technique of successive intertwinings of multiresolution analyses is demonstrated for the construction of piecewise-quadratic and piecewise-cubic multiwavelets.

4.3 Multiwavelets From Second-Order Finite Elements

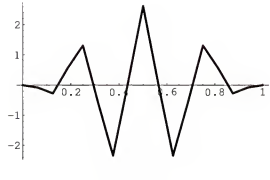
The principles discussed in the last section can be used to construct orthogonal multiresolution analyses from classical quadratic finite element spaces. When viewed as a finite shift-invariant space, the classical quadratic finite element spaces can be defined in terms of the two generators $\{\phi^1, \phi^2\}$ depicted in Figure (4.6). Following



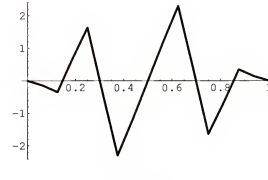
Wavelet ψ^1



Wavelet ψ^2



Wavelet ψ^3



Wavelet ψ^4

Figure 4.5: Piecewise-Linear Multiwavelets [2]

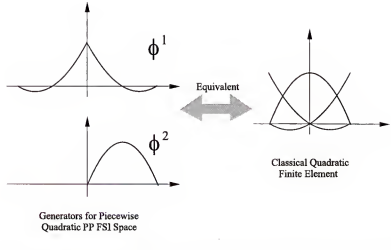


Figure 4.6: Quadratic Piecewise-Polynomials: FSI Space and FEM Basis

Table 4.1: Piecewise-Linear Scaling Filter Matrices [2]

$$[a_{-2}] = \begin{bmatrix} 0 & -\frac{-2\sqrt{2}+\sqrt{7}}{24\sqrt{6}} & -\frac{-65+14\sqrt{14}}{168\sqrt{6}} & -\frac{1}{14\sqrt{6}} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$[a_{-1}] = \begin{bmatrix} \frac{1}{14\sqrt{2}} & \frac{-2\sqrt{2}+\sqrt{7}}{24\sqrt{6}} & \frac{79+14\sqrt{14}}{168\sqrt{6}} & -\frac{13}{14\sqrt{6}} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$[a_0] = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{-2\sqrt{2}+\sqrt{7}}{24\sqrt{6}} & \frac{79+14\sqrt{14}}{168\sqrt{6}} & \frac{13}{14\sqrt{6}} \\ 0 & \frac{3}{4\sqrt{2}} & \frac{3}{4\sqrt{14}} & -\frac{1}{\sqrt{14}} \\ 0 & \frac{\sqrt{7}}{4} & -\frac{9}{28\sqrt{2}} & \frac{3}{7\sqrt{2}} \\ 0 & \frac{\sqrt{2}+4\sqrt{7}}{12\sqrt{2}} & -\frac{-4+\sqrt{14}}{12\sqrt{2}} & 0 \end{bmatrix}$$

$$[a_1] = \begin{bmatrix} \frac{1}{14\sqrt{2}} & \frac{4-\sqrt{14}}{48\sqrt{3}} & \frac{65\sqrt{2}-28\sqrt{7}}{336\sqrt{3}} & \frac{1}{14\sqrt{6}} \\ \sqrt{\frac{3}{14}} & \frac{3}{4\sqrt{2}} & \frac{3}{4\sqrt{14}} & \frac{1}{\sqrt{14}} \\ -\frac{3\sqrt{\frac{3}{2}}}{7} & \frac{\sqrt{\frac{7}{2}}}{4} & -\frac{9}{28\sqrt{2}} & -\frac{3}{7\sqrt{2}} \\ 0 & -\frac{1}{12}-\frac{\sqrt{\frac{7}{2}}}{3} & \frac{-2\sqrt{2}+\sqrt{7}}{12} & 0 \end{bmatrix}$$

Table 4.2: Piecewise-Linear Multiwavelet Filter Matrices [2]

$$[b_{-2}] = \begin{bmatrix} 0 & \frac{-2\sqrt{2}+\sqrt{7}}{6\sqrt{6}} & \frac{1+2\sqrt{14}}{6\sqrt{6}} & \frac{1}{\sqrt{6}} \\ \frac{2\sqrt{6}}{7} & 0 & -\frac{2\sqrt{2}}{7} & -\frac{3}{7\sqrt{2}} \\ -\frac{1}{14\sqrt{2}} & \frac{-2\sqrt{2}+\sqrt{7}}{24\sqrt{6}} & \frac{-65+14\sqrt{14}}{168\sqrt{6}} & -\frac{1}{14\sqrt{6}} \\ \frac{1}{14} & -\frac{-4+\sqrt{14}}{24\sqrt{6}} & -\frac{-65\sqrt{2}+28\sqrt{7}}{168\sqrt{6}} & \frac{1}{14\sqrt{3}} \end{bmatrix}$$

$$[b_{-1}] = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{14\sqrt{2}} & -\frac{-2\sqrt{2}+\sqrt{7}}{24\sqrt{6}} & -\frac{79+14\sqrt{14}}{168\sqrt{6}} & \frac{13}{14\sqrt{6}} \\ -\frac{1}{14} & -\frac{-4+\sqrt{14}}{24\sqrt{6}} & -\frac{79\sqrt{2}+28\sqrt{7}}{168\sqrt{6}} & \frac{13}{14\sqrt{3}} \end{bmatrix}$$

$$[b_0] = \begin{bmatrix} 0 & -\frac{-2\sqrt{2}+\sqrt{7}}{6\sqrt{6}} & -\frac{1+2\sqrt{14}}{6\sqrt{6}} & \frac{1}{\sqrt{6}} \\ 0 & 0 & -\frac{2\sqrt{2}}{7} & \frac{3}{7\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{-2\sqrt{2}+\sqrt{7}}{24\sqrt{6}} & -\frac{79+14\sqrt{14}}{168\sqrt{6}} & -\frac{13}{14\sqrt{6}} \\ 0 & \frac{-4+\sqrt{14}}{24\sqrt{6}} & \frac{79\sqrt{2}+28\sqrt{7}}{168\sqrt{6}} & \frac{13}{14\sqrt{3}} \end{bmatrix}$$

$$[b_1] = \begin{bmatrix} 0 & \frac{-2\sqrt{2}+\sqrt{7}}{6\sqrt{6}} & \frac{1+2\sqrt{14}}{6\sqrt{6}} & \frac{1}{\sqrt{6}} \\ \frac{2\sqrt{6}}{7} & 0 & -\frac{2\sqrt{2}}{7} & -\frac{3}{7\sqrt{2}} \\ -\frac{1}{14\sqrt{2}} & \frac{-2\sqrt{2}+\sqrt{7}}{24\sqrt{6}} & \frac{-65+14\sqrt{14}}{168\sqrt{6}} & \frac{1}{14\sqrt{6}} \\ \frac{1}{14} & -\frac{-4+\sqrt{14}}{24\sqrt{6}} & -\frac{-65\sqrt{2}+28\sqrt{7}}{168\sqrt{6}} & \frac{1}{14\sqrt{3}} \end{bmatrix}$$

the procedure outlined in the last section, we choose $\tilde{\phi}^3 \in V_1$ as depicted in Figure (4.7), where

$$\tilde{\phi}^3(x) = \phi^2(2x) - \phi^2(2x-1) \tag{4.17}$$

By inspection, $\tilde{\phi}^3$ is orthogonal to ϕ^2 and we simply choose $\tilde{\phi}^2 := \phi^2$. Both $\tilde{\phi}^2$ and $\tilde{\phi}^3$ are normalized to unit magnitude, and $\tilde{\phi}^1$ is constructed by applying the Gram-Schmidt orthogonalization procedure to ϕ^1:

$$\begin{aligned} \tilde{\phi}^1(x) &= \phi^1(x) - \left\langle \phi^1, \tilde{\phi}^2(\cdot+1) \right\rangle \tilde{\phi}^2(x+1) - \left\langle \phi^1, \tilde{\phi}^3(\cdot+1) \right\rangle \tilde{\phi}^3(x+1) \\ &\quad - \left\langle \phi^1, \tilde{\phi}^2 \right\rangle \tilde{\phi}^2(x) - \left\langle \phi^1, \tilde{\phi}^3 \right\rangle \tilde{\phi}^3(x) \end{aligned} \tag{4.18}$$

where $\langle \cdot, \cdot \rangle$ denotes the standard $L^2(\mathbb{R})$ inner product. In this fashion, a first intertwining multiresolution analysis is defined where $\tilde{V}_0$ is defined in terms of the generators $\left\{ \tilde{\phi}^1, \tilde{\phi}^2, \tilde{\phi}^3 \right\}$:

$$\tilde{V}_0 := span \left\{ \tilde{\phi}^s(\cdot - k) : \ s = 1, 2, 3 \right\}_{k \in \mathbb{Z}} \tag{4.19}$$

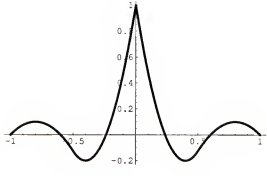
and, by construction,

$$V_0 \subset \tilde{V}_0 \subset V_1$$

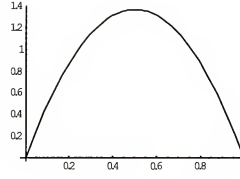
At this point, it is important to note that the spaces $\{\tilde{V}_j\}$ do not form an orthogonal multiresolution analysis. In fact, $\tilde{\phi}^1$ is not orthogonal to $\tilde{\phi}^1(\cdot - 1)$.

Proceeding to the second intertwining, a function w is selected as

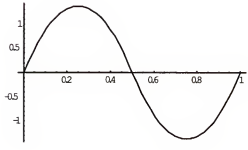
$$w(x) = \tilde{\phi}^1(2x-1) + a\left[\tilde{\phi}^2(2x) + \tilde{\phi}^2(2x-1)\right] + b\left[\tilde{\phi}^3(2x) - \tilde{\phi}^3(2x-1)\right] \tag{4.20}$$



Scaling Function ϕ^1



Scaling Function ϕ^2



Scaling Function ϕ^3

Figure 4.7: Generators for First Intertwining, Nonorthogonal Multiresolution Analysis

Recall that $A\left(\tilde{V}_1\right)$ is defined as the space spanned by all functions in $\tilde{V}_1$ whose support is contained in the interval $[0,1]$. Clearly, then, $w \in A(\tilde{V}_1)$. Moreover, w is symmetric and automatically orthogonal to $\tilde{\phi}^3$. Note that $A(\tilde{V}_0) = span\{\tilde{\phi}^2, \tilde{\phi}^3\}$. To apply Lemma (4.1), it is required that $w \in A(\tilde{V}_1) \ominus A(\tilde{V}_0)$. Therefore, one condition that must be satisfied by w is

$$\left\langle w, \tilde{\phi}^2 \right\rangle = 0 \tag{4.21}$$

The function w in equation (4.20) defines a one-dimensional space that is parameterized by two constants, a and b. An additional condition is needed to determine w, and this is provided by Lemma (4.1):

$$(I - P_W)\, C_0\left(\tilde{V}_0\right) \perp (I - P_W)\, C_1\left(\tilde{V}_0\right) \tag{4.22}$$

For the problem at hand, this condition is simple to impose. By the definitions of the functions $\left\{\tilde{\phi}^1, \tilde{\phi}^2, \tilde{\phi}^3\right\}$ and the spaces $C_0\left(\tilde{V}_0\right)$ and $C_1\left(\tilde{V}_0\right)$, we have

$$c_0(x) \;:=\; C_0\left(\tilde{V}_0\right) = \tilde{\phi}^1(x)\chi_{[0,1]}(x) \tag{4.23}$$

$$c_1(x) \;:=\; C_1\left(\tilde{V}_0\right) = \tilde{\phi}^1(x-1)\chi_{[0,1]}(x) \tag{4.24}$$

The orthogonality condition in equation (4.22) can then be written as

$$\langle (I - P_W)\, c_0, (I - P_W)\, c_1 \rangle = 0 \tag{4.25}$$

which can be expanded to

$$\langle c_0, c_1 \rangle + \langle P_W c_0, P_W c_1 \rangle - \langle c_0, P_W c_1 \rangle - \langle P_W c_0, c_1 \rangle = 0 \tag{4.26}$$

For any $f \in L^2(\mathbb{R})$, the orthogonal projection P_W is given by

$$P_W f = \frac{\langle w, f\rangle}{\|w\|}\frac{w}{\|w\|} = \frac{\langle w, f\rangle\, w}{\langle w, w\rangle} \tag{4.27}$$

Substituting the expression for the projection into equation (4.26), we obtain

$$\langle c_0, c_1\rangle + \left\langle \frac{\langle w, c_0\rangle\, w}{\langle w, w\rangle}, \frac{\langle w, c_1\rangle\, w}{\langle w, w\rangle} \right\rangle - \left\langle c_0, \frac{\langle w, c_1\rangle\, w}{\langle w, w\rangle} \right\rangle - \left\langle \frac{\langle w, c_0\rangle\, w}{\langle w, w\rangle}, c_1 \right\rangle = 0$$

which reduces to

$$\langle w, w\rangle\, \langle c_0, c_1\rangle - \langle w, c_0\rangle\, \langle w, c_1\rangle = 0 \tag{4.28}$$

With the solution of equations (4.21) and (4.28) for the constants a and b in equation (4.20), the function w is then normalized. Finally, Lemma (4.1) guarantees that the translates of $\tilde{\phi}^1$ will be mutually orthogonal if $\tilde{\phi}^1$ is orthogonal to the translates of w. Therefore, applying the Gram-Schmidt procedure, we obtain

$$\tilde{\tilde{\phi}}^1(x) = \tilde{\phi}^1(x) - \left\langle \tilde{\phi}^1, w(\cdot + 1)\right\rangle w(x+1) - \left\langle \tilde{\phi}^1, w\right\rangle w(x) \tag{4.29}$$

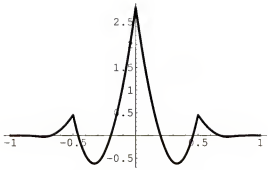
The function $\tilde{\tilde{\phi}}^1$ is then normalized and we define

$$\begin{aligned} \tilde{\tilde{\phi}}^2 &:= \tilde{\phi}^2 \\ \tilde{\tilde{\phi}}^3 &:= \tilde{\phi}^3 \\ \tilde{\tilde{\phi}}^4 &:= w \end{aligned}$$

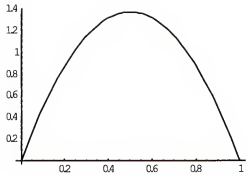
Then, the final orthogonal multiresolution analysis $\{\tilde{\tilde{V}}_j\}_{j\in\mathbb{Z}}$ is obtained, where

$$\tilde{\tilde{V}}_0 := span\left\{\tilde{\tilde{\phi}}^s(\cdot - k) :\ s = 1, \ldots, 4\right\}_{k\in\mathbb{Z}} \tag{4.30}$$

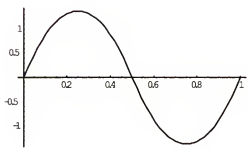
Now that an orthogonal multiresolution analysis has been generated using the technique of interwining, the double tilde notation is dropped for the multiresolution



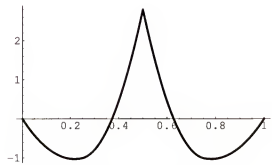
Scaling Function ϕ^1



Scaling Function ϕ^2



Scaling Function ϕ^3



Scaling Function ϕ^4

Figure 4.8: Generators for Final Intertwining, Orthogonal Multiresolution Analysis

analysis defined in equation (4.30). The generators $\{\phi^1, \phi^2, \phi^3, \phi^4\}$ are depicted in Figure (4.8). The approximation spaces $\{V_j\}$ are then defined as

$$V_j := span\left\{\phi^s_{j,k}(x) : \ s = 1, \ldots, 4\right\}_{k \in \mathbb{Z}} \tag{4.31}$$

where

$$\phi^s_{j,k}(x) = 2^{j/2}\phi^s\left(2^j x - k\right) \tag{4.32}$$

The scaling filter matrices associates with these generators are listed in Table (4.3).

Table 4.3: Piecewise-Quadratic Scaling Filter Matrices

n	$[a_n]$			
-2	0	.0163446406689	-.0342630216987	-.0327571725062
	0	0	0	0
	0	0	0	0
	0	0	0	0
-1	.113474174186	.0163446406689	-.422172442890	-.236881317738
	0	0	0	0
	0	0	0	0
	0	0	0	0
0	.707106781187	.0163446406689	.422172442890	-.236881317738
	0	.618718433538	-.220970869121	-.0988211768803
	0	.707106781187	0	0
	0	-.338665211381	-.316332942563	-.201862521835
1	.113474174186	.0163446406689	.0342630216987	-.0327571725062
	.342326598441	.618718433538	.220970869121	-.0988211768803
	0	-.707106781187	0	0
	.699272287923	-.338665211381	.316332942563	-.201862521835

Now that a set of orthonormal scaling functions has been constructed, it is not difficult to derive an associated set of wavelets. Clearly, from the two-scale equations, each wavelet must be composed of a linear combination of scaling functions that span V_1. Furthermore, the space W_0 is the orthogonal complement to V_0, so the wavelets must be constructed so that they are orthogonal to all translates of the scaling functions $\{\phi^1, \phi^2, \phi^3, \phi^4\}$.

First, a symmetric wavelet, supported over $[-1, 1]$, is constructed. We choose

$$\psi^1(x) = \phi^1_{1,0}(x) - \langle \phi^1_{1,0}, \phi^1 \rangle \, \phi^1(x) \tag{4.33}$$

Note that the Gram-Schmidt procedure has been used to construct ψ^1 such that it is orthogonal to ϕ^1. By definition in equation (4.32), the function $\phi^1_{1,0}$ is a scaled and contracted form of ϕ^1, also symmetric about the origin and supported over $[-\frac{1}{2}, \frac{1}{2}]$. Note that, because the remaining scaling functions $\{\phi^2, \phi^3, \phi^4\}$ are supported over $[0, 1]$, there is no component of $\phi^1_{1,0}$ in these functions. Consequently, they are orthogonal to $\phi^1_{1,0}$. The symmetric wavelet ψ^1 is then normalized.

Next, an antisymmetric wavelet that is supported on $[-1, 1]$ is constructed. We start with a symmetric function

$$z(x) = \phi^1(x) - \langle \phi^1, \phi^1_{1,0} \rangle \, \phi^1_{1,0}(x) \tag{4.34}$$

The contribution of $\phi^1_{1,0}$ is removed from ϕ^1 in order to obtain a function that passes through the origin. The function z is made antisymmetric by multiplying by the function H, defined as

$$H(x) := \begin{cases} -1 & x \in [-1, 0) \\ 1 & x \in [0, 1] \\ 0 & \text{otherwise} \end{cases} \tag{4.35}$$

We obtain

$$\psi^2(x) = H(x)z(x) = H(x) \left[\phi^1(x) - \langle \phi^1, \phi^1_{1,0} \rangle \, \phi^1_{1,0}(x) \right] \tag{4.36}$$

Since ψ^2 is an antisymmetric function, it is guaranteed to be orthogonal to ϕ^1 and ψ^1. By construction, the translates of the functions ψ^2, ψ^1, and ϕ^1 are all mutually orthogonal. Clearly, ψ^2 is also orthogonal to the remaining scaling functions $\{\phi^2, \phi^3, \phi^4\}$ since both ϕ^1 and $\phi^1_{1,0}$ are orthogonal to these functions.

In order to complete the set of wavelets, two wavelets that are supported over $[0, 1]$ are constructed. Recall that these wavelets must be formed as linear combinations of the scaling functions that span V_1 and have support contained in $[0, 1]$. This implies that the remaining wavelets take the form

$$\begin{aligned} \psi^t(x) &= a\phi^2_{1,0}(x) + b\phi^3_{1,0}(x) + c\phi^4_{1,0}(x) + d\phi^2_{1,1}(x) \\ &\quad + e\phi^3_{1,1}(x) + f\phi^4_{1,1}(x) + g\phi^1_{1,1}(x), \quad t = 3, 4 \end{aligned} \tag{4.37}$$

where the seven constants $\{a, b, c, d, e, f, g\}$ must be determined. The wavelets must be constructed such that they are orthogonal to the translates of the scaling functions $\{\phi^1, \phi^2, \phi^3, \phi^4\}$. This requirement provides five orthogonality constraints for ψ^t, $t = 3, 4$:

$$\langle \psi^t, \phi^s \rangle = 0, \quad s = 1, \ldots, 4 \tag{4.38}$$
$$\langle \psi^t, \phi^1(\cdot - 1) \rangle = 0$$

Note that since there are five orthogonality conditions and seven unknown constants, there is some freedom in the design of ψ^3 and ψ^4 as long as it is ensured that they are linearly independent. Finally, the Gram-Schmidt procedure is used to make the two functions orthogonal. In this case, ψ^3 has been constructed to be an antisymmetric function and ψ^4 has been constructed to be a symmetric function.

We now have a set of orthonormal wavelets $\{\psi^1, \psi^2, \psi^3, \psi^4\}$ whose translates form a basis for the orthogonal complement space W_0. These wavelets are depicted in Figure (4.9). The wavelet spaces $\{W_j\}$ are defined as

$$W_j := span\left\{\psi^s_{j,k} : \ s = 1, \ldots, 4\right\}_{k \in \mathbb{Z}} \tag{4.39}$$

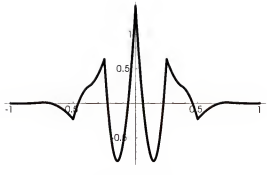
where

$$\psi^s_{j,k}(x) = 2^{j/2}\psi^s\left(2^j x - k\right) \tag{4.40}$$

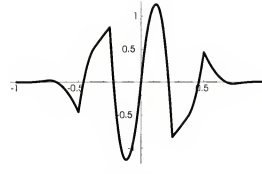
Table (4.4) lists the wavelet filter matrices associated with these piecewise-quadratic multiwavelets.

4.4 Multiwavelets From Third-Order Finite Elements

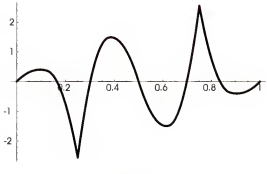
Using the same procedure that was applied in the quadratic case, an orthogonal multiresolution analysis can be constructed from the classical cubic finite element



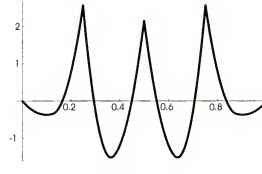
Wavelet ψ^1



Wavelet ψ^2



Wavelet ψ^3



Wavelet ψ^4

Figure 4.9: Piecewise-Quadratic Multiwavelets

Table 4.4: Piecewise-Quadratic Multiwavelet Filter Matrices

n	$[b_n]$			
-2	0	-.0163446406689	.0342630216987	.0327571725062
	0	-.0231148125061	.0484552299729	.0463256376237
	0	0	0	0
	0	0	0	0
-1	-.113474174186	-.0163446406689	.422172442890	.236881317738
	-.160476716113	-.0231148125061	.597041994395	.335000772219
	0	0	0	0
	0	0	0	0
0	.707106781187	-.0163446406689	-.422172442890	.236881317738
	0	.0231148125061	.597041994395	-.335000772219
	0	0	-.288675134594	-.645497224366
	0	.0188731653801	.223959608749	.551430926265
1	-.113474174186	-.0163446406689	-.0342630216987	.0327571725062
	.160476716113	.0231148125061	.0484552299729	-.0463256376237
	0	0	-.288675134594	.645497224366
	.539276980493	.0188731653801	-.223959608749	.551430926265

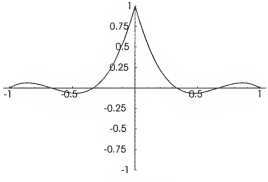
spaces. When viewed as finite shift-invariant spaces, the classical cubic finite element spaces are defined in terms of the three generators $\{\phi^1, \phi^2, \phi^3\}$ depicted in Figure (4.10). Because it is more convenient to work with symmetric and antisymmetric functions, an alternate set of generators $\{\phi^1, \varphi^2, \varphi^3\}$ is chosen, where

$$\varphi^2 = \phi^2 + \phi^3 \tag{4.41}$$

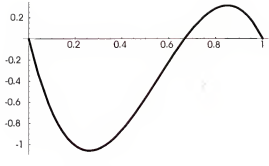
$$\varphi^3 = \phi^2 - \phi^3 \tag{4.42}$$

These functions, shown in Figure (4.11), are orthogonal due to symmetry. Clearly, since φ^2 and φ^3 are both formed as linear combinations of ϕ^2 and ϕ^3, $\{\phi^1, \varphi^2, \varphi^3\}$ is also a basis for the classical cubic finite element spaces.

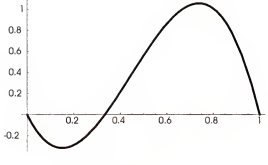
Proceeding to the first intertwining multiresolution analysis, the functions φ^2 and φ^3 are normalized and we define



Scaling Function ϕ^1

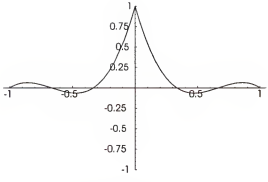


Scaling Function ϕ^2

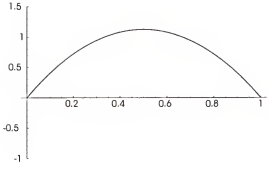


Scaling Function ϕ^3

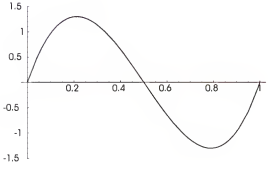
Figure 4.10: Generators for the Classical Cubic Finite Element Spaces



Scaling Function ϕ^1



Scaling Function φ^2



Scaling Function φ^3

Figure 4.11: Alternate Generators for the Cubic Finite Element Spaces

$$\tilde{\phi}^2 := \varphi^2$$
$$\tilde{\phi}^3 := \varphi^3$$

Next, a function ϕ^4 is selected that is a linear combination of the functions that span $A(V_1)$. We choose ϕ^4 to be an antisymmetric function of the form

$$\phi^4 = \left[\tilde{\phi}^2(2x) - \tilde{\phi}^2(2x-1)\right] + \left[\tilde{\phi}^3(2x) + \tilde{\phi}^3(2x-1)\right] \tag{4.43}$$

By symmetry, ϕ^4 is orthogonal to $\tilde{\phi}^2$. Then, ϕ^4 is made orthogonal to $\tilde{\phi}^3$ by applying the Gram-Schmidt orthogonalization procedure:

$$\tilde{\phi}^4(x) = \phi^4(x) - \left\langle \phi^4, \tilde{\phi}^3 \right\rangle \tilde{\phi}^3(x) \tag{4.44}$$

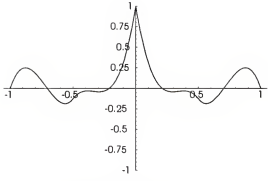
The function $\tilde{\phi}^4$ is normalized and $\tilde{\phi}^1$ is constructed using the Gram-Schmidt procedure:

$$\begin{aligned} \tilde{\phi}^1(x) &= \phi^1(x) - \left\langle \phi^1, \tilde{\phi}^2(\cdot+1) \right\rangle \tilde{\phi}^2(x+1) - \left\langle \phi^1, \tilde{\phi}^3(\cdot+1) \right\rangle \tilde{\phi}^3(x+1) \\ &\quad - \left\langle \phi^1, \tilde{\phi}^4(\cdot+1) \right\rangle \tilde{\phi}^4(x+1) - \left\langle \phi^1, \tilde{\phi}^2 \right\rangle \tilde{\phi}^2(x) \\ &\quad - \left\langle \phi^1, \tilde{\phi}^3 \right\rangle \tilde{\phi}^3(x) - \left\langle \phi^1, \tilde{\phi}^4 \right\rangle \tilde{\phi}^4(x) \end{aligned} \tag{4.45}$$

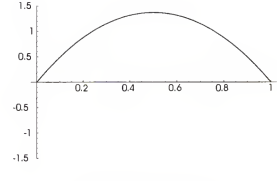
A first intertwining multiresolution analysis $\{\tilde{V}_j\}$ can now be defined in terms of the generators $\left\{\tilde{\phi}^1, \tilde{\phi}^2, \tilde{\phi}^3, \tilde{\phi}^4\right\}$, where the space $\tilde{V}_0$ is defined as

$$\tilde{V}_0 := span\left\{\tilde{\phi}^s(\cdot - k) : \ s = 1, \ldots, 4\right\}_{k \in \mathbb{Z}} \tag{4.46}$$

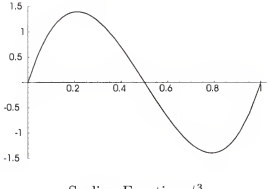
The generators for the first intertwining multiresolution analysis are shown in Figure (4.12). Note that, just as in the quadratic case, this first intertwining does not



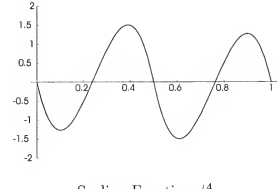
Scaling Function ϕ^1



Scaling Function ϕ^2



Scaling Function ϕ^3



Scaling Function ϕ^4

Figure 4.12: Generators for the First Intertwining, Nonorthogonal Multiresolution Analysis

generate an orthogonal multiresolution analysis since the translates of $\tilde{\phi}^1$ are not mutually orthogonal.

Proceeding to the second intertwining, a symmetric function $w \in A\left(\tilde{V}_1\right)$ is defined. This function is chosen to take the form

$$w(x) = \tilde{\phi}^1(2x-1) + a\left[\tilde{\phi}^2(2x) + \tilde{\phi}^2(2x-1)\right] + b\left[\tilde{\phi}^3(2x) - \tilde{\phi}^3(2x-1)\right] \qquad (4.47)$$

Because w is symmetric, it is automatically orthogonal to the antisymmetric functions $\tilde{\phi}^3$ and $\tilde{\phi}^4$. It is required that w also be orthogonal to $\tilde{\phi}^2$, so w must satisfy the constraint

$$\left\langle w, \tilde{\phi}^2 \right\rangle = 0 \tag{4.48}$$

If equation (4.48) is satisfied, then $w \in A(\tilde{V}_1) \ominus A(\tilde{V}_0)$. Recall that Lemma (4.1) provides an additional condition that w must satisfy. We require

$$(I - P_W)\, C_0\left(\tilde{V}_0\right) \perp (I - P_W)\, C_1\left(\tilde{V}_0\right) \tag{4.49}$$

Similar to the quadratic case, the spaces $C_0(\tilde{V}_0)$ and $C_1(\tilde{V}_0)$ are one-dimensional spaces defined as

$$c_0(x) := C_0\left(\tilde{V}_0\right) = \tilde{\phi}^1(x)\chi_{[0,1]}(x) \tag{4.50}$$

$$c_1(x) := C_1\left(\tilde{V}_0\right) = \tilde{\phi}^1(x-1)\chi_{[0,1]}(x) \tag{4.51}$$

Just as in the quadratic case, the orthogonality condition in equation (4.49) can be reduced to

$$\langle w, w\rangle \langle c_0, c_1\rangle - \langle w, c_0\rangle \langle w, c_1\rangle = 0 \tag{4.52}$$

Equations (4.48) and (4.52) can be solved for the unknown constants a and b in equation (4.47). Then, w is normalized. Finally, Lemma (4.1) guarantees that an orthogonal multiresolution analysis will be obtained if it is ensured that $\tilde{\phi}^1$ is orthogonal to the translates of w. Towards this end, the Gram-Schmidt procedure is applied once again:

$$\tilde{\tilde{\phi}}^1(x) = \tilde{\phi}^1(x) - \left\langle \tilde{\phi}^1, w(\cdot + 1) \right\rangle w(x+1) - \left\langle \tilde{\phi}^1, w \right\rangle w(x) \tag{4.53}$$

The function $\tilde{\tilde{\phi}}^1$ is then normalized and we define

$$\begin{aligned}
\tilde{\tilde{\phi}}^2 &:= \tilde{\phi}^2 \\
\tilde{\tilde{\phi}}^3 &:= \tilde{\phi}^3 \\
\tilde{\tilde{\phi}}^4 &:= \tilde{\phi}^4 \\
\tilde{\tilde{\phi}}^5 &:= w
\end{aligned}$$

Then, the final orthogonal multiresolution analysis $\{\tilde{\tilde{V}}_j\}_{j\in Z}$ is obtained, where

$$\tilde{\tilde{V}}_0 := span\left\{\tilde{\tilde{\phi}}^s(\cdot - k) : \ s = 1, \ldots, 5\right\}_{k\in\mathbb{Z}} \tag{4.54}$$

For convenience, the double tilde notation is dropped for the multiresolution analysis defined in equation (4.54). The generators $\{\phi^1, \phi^2, \phi^3, \phi^4, \phi^5\}$ are depicted in Figure (4.13). The approximation spaces $\{V_j\}$ are then defined as

$$V_j := span\left\{\phi^s_{j,k} : \ s = 1, \ldots, 5\right\}_{k\in\mathbb{Z}} \tag{4.55}$$

where, as before,

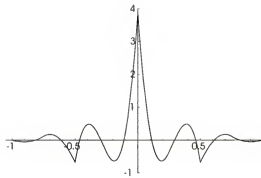
$$\phi^s_{j,k}(x) = 2^{j/2}\phi^s\left(2^j x - k\right) \tag{4.56}$$

The scaling filter matrices associates with these generators are listed in Table (4.4).

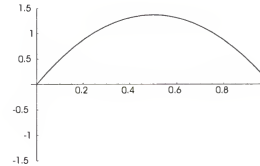
It now remains to construct a set of orthonormal wavelets associated with the above orthogonal multiresolution analysis. This construction follows closely the procedure used to derive the quadratic wavelets. As in the quadratic case, a symmetric wavelet, supported over $[-1, 1]$, is constructed:

$$\psi^1(x) = \phi^1_{1,0}(x) - \left\langle \phi^1_{1,0}, \phi^1 \right\rangle \phi^1(x) \tag{4.57}$$

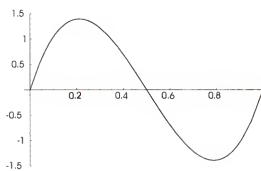
This function is orthogonal to the scaling functions $\{\phi^1, \phi^2, \phi^3, \phi^4\, \phi^5\}$ by the same argument employed in the quadratic case. Then, ψ^1 is normalized and we proceed



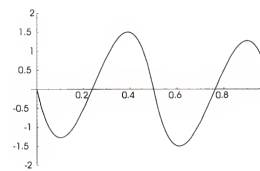
Scaling Function ϕ^1



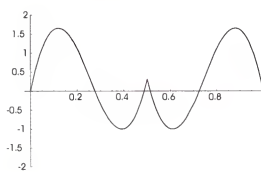
Scaling Function ϕ^2



Scaling Function ϕ^3



Scaling Function ϕ^4



Scaling Function ϕ^5

Figure 4.13: Generators for the Final Intertwining, Orthogonal Multiresolution Analysis

Table 4.5: Piecewise-Cubic Scaling Filter Matrices

n	$[a_n]$				
-2	0	.0187976275001	.00522145045609	-.0611853406101	-.0758597160812
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
-1	-.123878397621	.0187976275001	-.00522145045609	.311185340610	.357152985811
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
0	.707106781187	.0187976275001	.00522145045609	-.311185340610	.357152985811
	0	.618718433538	-.233853586673	.0883883476483	.153093108924
	0	.701560760020	.0883883476483	0	0
	0	.0883883476483	-.701560760020	0	0
	0	.224458784373	.666550563194	.0488842065142	.0354900499726
1	-.123878397621	.0187976275001	-.00522145045609	.0611853406101	-.0758597160812
	.250000000000	.618718433538	.233853586673	-.0883883476483	.153093108924
	0	-.701560760020	.0883883476483	0	0
	0	-.0883883476483	-.701560760020	0	0
	.057955089191	.224458784373	-.666550563194	-.0488842065142	.0354900499726

to the construction of an antisymmetric wavelet that is supported over $[-1, 1]$. As in the quadratic case, we start with the symmetric function

$$z(x) = \phi^1(x) - \left\langle \phi^1, \phi^1_{1,0} \right\rangle \phi^1_{1,0}(x) \tag{4.58}$$

Then, an antisymmetric function is generated by multiplying z by the function H, defined in equation (4.41), to obtain

$$\psi^2(x) = H(x)z(x) = H(x)\left[\phi^1(x) - \left\langle \phi^1, \phi^1_{1,0} \right\rangle \phi^1_{1,0}(x)\right] \tag{4.59}$$

By construction, this function is orthogonal to the scaling functions $\{\phi^1, \phi^2, \phi^3, \phi^4\, \phi^5\}$ and their translates. Furthermore, it is also orthogonal to all translates of the symmetric wavelet ψ^1.

Finally, three wavelets that are supported over $[0, 1]$ are constructed. These wavelets are formed as linear combinations of the scaling functions that span V_1 and have support contained in $[0, 1]$. Then, these wavelets take the form

$$\psi^t(x) \;=\; a\phi^2_{1,0}(x) + b\phi^3_{1,0}(x) + c\phi^4_{1,0}(x) + d\phi^5_{1,0}(x) + e\phi^2_{1,1}(x) \tag{4.60}$$

$$+f\phi^3_{1,1}(x) + g\phi^4_{1,1}(x) + h\phi^5_{1,1}(x) + i\phi^1_{1,1}(x), \qquad t = 3, 4, 5$$

Equation (4.60) contains nine constants that must be determined. Because the wavelets must be constructed such that they are orthogonal to the translates of the scaling functions $\{\phi^1, \phi^2, \phi^3, \phi^4, \phi^5\}$, we have the following six orthogonality constraints for ψ^t, $t = 3, 4, 5$:

$$\begin{aligned} \langle \psi^t, \phi^s \rangle &= 0, \qquad s = 1, \ldots, 5 \\ \langle \psi^t, \phi^1(\cdot - 1) \rangle &= 0 \end{aligned} \tag{4.61}$$

Clearly, we have the freedom to choose three of the constants for each of the wavelets in equation (4.60) as long as the resulting functions are linearly independent. The Gram-Schmidt procedure is used to ensure that these wavelets are also mutually orthogonal. In this case, ψ^3 and ψ^5 have been chosen to be symmetric functions while ψ^4 is chosen to be an antisymmetric function.

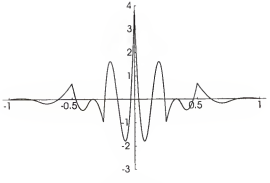
We now have a set of piecewise-cubic, orthonormal wavelets $\{\psi^1, \psi^2, \psi^3, \psi^4\, \psi^5\}$ whose translates form a basis for the orthogonal complement space W_0. These wavelets are depicted in Figure (4.14). The wavelet spaces $\{W_j\}$ are defined as

$$W_j := span\left\{\psi^s_{j,k} : \ s = 1, \ldots, 5\right\}_{k \in \mathbb{Z}} \tag{4.62}$$

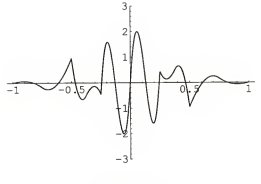
where, as usual,

$$\psi^s_{j,k}(x) = 2^{j/2}\psi^s\left(2^j x - k\right) \tag{4.63}$$

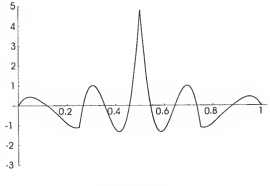
The wavelet filter matrices associated with these piecewise-cubic multiwavelets are listed in Table (4.4).



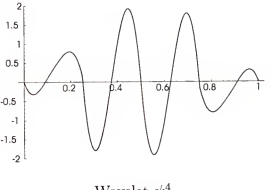
Wavelet ψ^1



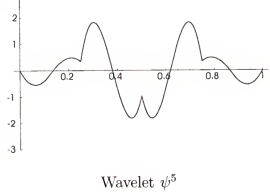
Wavelet ψ^2



Wavelet ψ^3



Wavelet ψ^4



Wavelet ψ^5

Figure 4.14: Piecewise-Cubic Multiwavelets

Table 4.6: Piecewise-Cubic Multiwavelet Filter Matrices

n			$[b_n]$			
-2	0	-.0187976275001	-.0052215045609	.0611853406101	.0758597160812
	0	-.0265838597511	-.0073842605026	.0865291385093	.107281839320
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
-1	.123878397621	-.0187976275001	.0052215045609	-.311185340610	-.357152985811
	.175190510001	-.0265838597511	.0073842605026	-.440082529103	-.505090596376
	0	0	0	0	0
	0	0	0	0	0
	0	0	0	0	0
0	.707106781187	-.0187976275001	-.0052215045609	.311185340610	-.357152985811
	0	.0265838597511	.0073842605026	-.440082529103	.505090596376
	0	-.124934166510	.0260907284726	-.305772920466	0
	0	0	0	.612372435696	.353553390593
	0	.219932720984	-.0111360780385	-.344936091180	-.563072951901
1	.123878397621	-.0187976275001	.0052215045609	-.0611853406101	.0758597160812
	-.175190510001	.0265838597511	-.0073842605026	.0865291385093	-.107281839320
	.883417963408	-.124934166510	-.0260907284726	.305772920466	0
	0	0	0	.612372435696	-.353553390593
	-.175917764003	.219932720984	.0111360780385	.344936091180	-.563072951901

4.5 Piecewise-Polynomial Multiwavelets of Arbitrary Order

In the previous two sections, the construction of orthogonal, piecewise-quadratic and piecewise-cubic multiwavelets has been demonstrated using successive intertwining multiresolution analyses. This procedure can be applied to generate higher-order, orthogonal multiresolution analyses from the classical (Lagrangian) finite element basis functions of appropriate order. The following theorem can be stated regarding the construction of orthogonal, piecewise-polynomial multiwavelets of arbitrary order:

Theorem 4.1: An orthogonal multiresolution analysis of order r, $r > 1$, can be generated from the classical (Lagrangian) finite element spaces of the same order. The scaling functions and wavelets that correspond to the resulting multiresolution analysis possess the following properties:

(1) The multiresolution analysis is generated by $r+2$ orthogonal scaling functions of order r, minimally supported over $[-1, 1]$. One scaling function, ϕ^1, is symmetric about the origin and supported over $[-1, 1]$. The remaining $r+1$ scaling functions, $\{\phi^2, \ldots, \phi^{r+2}\}$, are supported over $[0, 1]$.

(2) There are $r+2$ corresponding orthogonal wavelets, minimally supported over $[-1, 1]$. One wavelet, ψ^1, is symmetric about the origin and supported over

$[-1, 1]$. A second wavelet, ψ^2, is antisymmetric about the origin and supported over $[-1, 1]$. The remaining r wavelets, $\{\psi^3, \ldots, \psi^{r+2}\}$, are supported over $[0, 1]$.

Proof: (1) The classical finite element spaces of order r are generated by $r + 1$ Lagrangian interpolation functions of order r. Two of these functions are nonzero at the element boundary, while the remaining $r - 1$ "interior" functions are zero at the boundaries. When these spaces are viewed as a finite shift-invariant spaces, the two functions that are nonzero at the element boundaries can be assembled into one symmetric function, spanning two elements. If the length of the standard finite element is taken as one, a shift-invariant space is generated by one function supported over $[-1, 1]$ and $r - 1$ interior functions supported over $[0, 1]$. As discussed earlier, an orthogonal multiresolution analysis of order r can be generated from this shift-invariant space using the technique of successive intertwinings. Each intertwining involves the construction of an additional scaling function, supported over $[0, 1]$. Therefore, the final, orthogonal multiresolution analysis will have $r + 2$ generators, one supported over $[-1, 1]$ and $r + 1$ supported over $[0, 1]$.

(2) Part (2) of the above theorem follows as a special case of Lemmas 4.1, 4.2, and 4.4 [3]. The following argument is a simplified form of the more general proof provided by Donovan, Geronimo, and Hardin in Ref. [3]. First, it is demonstrated that a symmetric wavelet can be constructed with support over $[-1, 1]$. Consider the function

$$\psi^1(x) = \phi^1_{1,0}(x) - \langle \phi^1, \phi^1_{1,0} \rangle \, \phi^1(x) \tag{4.64}$$

supported over $[-1, 1]$. Recall from equation (4.56) that $\phi^1_{1,0}$ is a scaled, contracted form of ϕ^1. Because ϕ^1 and $\phi^1_{1,0}$ are symmetric about the origin, so is ψ^1. Clearly, $\psi^1 \in V_1$ since $\phi^1_{1,0} \in V_1$ and $\phi^1 \in V_0 \subset V_1$. By construction, ψ^1 is orthogonal to ϕ^1. Because $\phi^1_{1,0}$ is supported over $[-\frac{1}{2}, \frac{1}{2}]$, and the scaling functions $\{\phi^2, \ldots, \phi^{r+2}\}$ are supported over $[0, 1]$, it is clear that $\{\phi^2, \ldots, \phi^{r+2}\}$ contain no component of $\phi^1_{1,0}$.

Therefore, ψ^1 is also orthogonal to the scaling functions $\{\phi^2, \ldots, \phi^{r+2}\}$, and it can be concluded that $\psi^1 \in V_1 \ominus V_0$. Recall, from the definition of the wavelet complement spaces,

$$W_0 := V_1 \ominus V_0 \tag{4.65}$$

Clearly, then, $\psi^1 \in W_0$ and we have proven the existence of a symmetric wavelet that is supported over $[-1, 1]$.

Next, it can be shown that an antisymmetric wavelet can be constructed with support over $[-1, 1]$. We start with the function

$$z(x) = \phi^1(x) - \left\langle \phi^1, \phi^1_{1,0} \right\rangle \phi^1_{1,0}(x) \tag{4.66}$$

The symmetric function z is orthogonal to the scaling functions $\{\phi^2, \ldots, \phi^{r+2}\}$. Furthermore, by removing the contribution of $\phi^1_{1,0}$ to ϕ^1, the resulting function z passes through the origin. An antisymmetric function is created by multiplying z by the function H, defined in equation (4.41). We obtain

$$\psi^2(x) = z(x)H(x) \tag{4.67}$$

By symmetry, ψ^2 is orthogonal to the symmetric functions ϕ^1 and ψ^1. It can also be shown that, by construction, ψ^2 is also orthogonal to the translates of ϕ^1 and ψ^1. Then, it can be concluded that $\psi^2 \in W_0$ and we have proven the existence of an antisymmetric wavelet that is supported over $[-1, 1]$.

Finally, it remains to be shown that there are exactly r wavelets supported over $[0, 1]$. First, note that these interior wavelets can be composed only of functions that are contained in the space $A(V_1)$. Recall that $A(V_1)$ is defined as the space spanned by the basis functions of V_1 whose support is contained in $[0, 1]$. Therefore, $A(V_1)$ is defined as

$$A(V_1) := span\left\{\phi_{1,1}^1, \phi_{1,0}^2, \ldots, \phi_{1,0}^{r+2}, \phi_{1,1}^2, \ldots, \phi_{1,1}^{r+2}\right\} \tag{4.68}$$

and the dimension of $A(V_1)$ is $2r+3$. The wavelets must be orthogonal to the scaling functions that span $A(V_0)$, defined as the $r+1$-dimensional space

$$A(V_0) := span\left\{\phi^2, \ldots, \phi^{r+2}\right\} \tag{4.69}$$

In addition, the wavelets must be constructed such that they are orthogonal to the translates of ϕ^1, the scaling function supported over $[-1, 1]$. We define the spaces

$$Q_0 := span\left\{\phi^1 \chi_{[0,1]}\right\} \tag{4.70}$$

$$Q_1 := span\left\{\phi^1(\cdot - 1)\chi_{[0,1]}\right\} \tag{4.71}$$

That is, Q_0 and Q_1 are the one-dimensional spaces spanned by the restrictions of ϕ^1 and $\phi^1(\cdot - 1)$ to $[0, 1]$. Defining Ψ^i to be the space spanned by the wavelets that are supported over $[0, 1]$, we have

$$\Psi^i = A(V_1) \ominus A(V_0) \ominus Q_0 \ominus Q_1 \tag{4.72}$$

since $\Psi^i \subset A(V_1)$ and must be orthogonal to the spaces $A(V_0)$, Q_0, and Q_1. The dimension of Ψ^i is then calculated to be

$$\begin{aligned} dim(\Psi^i) &= dim\left(A(V_1)\right) - dim\left(A(V_0)\right) - dim\left(Q_0\right) - dim\left(Q_1\right) \\ &= (2r+3) - (r+1) - 1 - 1 \\ &= r \end{aligned}$$

Therefore, Ψ^i is an r-dimensional space and there must be r wavelets supported over $[0, 1]$. This completes the proof.

The above theorem guarantees that we can generate an orthogonal multiresolution analysis of arbitrary order greater than one. First-order, linear multiwavelets are an exception to the theorem because the resulting orthogonal multiresolution analysis has four scaling functions and wavelets, while the theorem would imply that there are only three. The extra scaling function and wavelet are needed because three dimensions are not enough to construct an orthogonal multiresolution analysis from the classical linear finite element basis functions. As discussed earlier, this construction has been carried out by Donovan, Geronimo, and Hardin [2]. In the previous two sections, the applicability of the above theorem has been demonstrated for piecewise-quadratic and piecewise-cubic multiwavelets. The scaling functions are generated using the technique of successive intertwining multiresolution analyses, while the wavelets are constructed according to the procedure outlined in the above proof. The significance of the theorem is that these methods can be applied to generate orthogonal multiresolution analyses of arbitrary approximation order.

4.6 Boundary-Adapted Multiwavelets

It is not difficult to adapt the piecewise-polynomial multiwavelets that have been derived from classical finite element spaces to finite boundaries. This is necessary so that the multiwavelets can be used to approximate functions that are supported over finite domains. In particular, this will prove useful later when these multiwavelets are employed in the representation of first and second-order Volterra kernels, which have finite support.

As an example, consider the case in which a wavelet basis is needed for $L^2[0,1]$, the space of square-integrable functions supported over $[0,1]$. This case can easily be generalized to other one-dimensional domains. In this example, the piecewise-linear multiwavelets are used although the results are applicable to the higher-order multiwavelets as well. In constructing a basis for $L^2[0,1]$ using the piecewise-linear multiwavelets, recall that there are a total of four scaling functions $\{\phi^1, \phi^2, \phi^3, \phi^4\}$

defined on level 0. The function ϕ^1 is supported over $[-1,1]$ while the functions $\{\phi^2, \phi^3, \phi^4\}$ are all supported over $[0,1]$. The space $V_0 \subset L^2(\mathbb{R})$ is defined as the span of all integer translates of these four scaling functions. In constructing a basis for $V_0[0,1] \subset L^2[0,1]$, note that there are a total of five functions that intersect the domain $[0,1]$, as shown in Figure (4.15). From the figure, it is clear that the functions ϕ^1 and $\phi^1(\cdot - 1)$ both overlap the boundaries. The functions $\phi^1\chi_{[0,1]}$ and $\phi^1(\cdot - 1)\chi_{[0,1]}$ are defined as the restrictions of these functions to the domain $[0,1]$:

$$\phi^1(x)\chi_{[0,1]}(x) := \begin{cases} \phi^1(x) & x \in [0,1] \\ 0 & \text{otherwise} \end{cases} \tag{4.73}$$

$$\phi^1(x-1)\chi_{[0,1]}(x) := \begin{cases} \phi^1(x-1) & x \in [0,1] \\ 0 & \text{otherwise} \end{cases} \tag{4.74}$$

where $\chi_{[0,1]}$ is the characteristic function over $[0,1]$. These functions are depicted in Figure (4.16). Then, the space $V_0[0,1]$ is defined as

$$V_0[0,1] := span\left\{\phi^1\chi_{[0,1]}, \phi^2, \phi^3, \phi^4, \phi^1(\cdot - 1)\chi_{[0,1]}\right\} \tag{4.75}$$

In a similar manner, on resolution level 1 there are a total of nine scaling functions that intersect the domain $[0,1]$. These functions are shown in Figure (4.17) where, as in previous sections, we have the notation

$$\phi^s_{j,k}(x) = 2^{j/2}\phi^s(2^j x - k) \tag{4.76}$$

The functions $\phi^1_{1,0}$ and $\phi^1_{1,2}$ overlap the boundaries. Restricting these functions to $[0,1]$, the space $V_1[0,1]$ is defined as

$$V_1[0,1] := span\left\{\phi^1_{1,0}\chi_{[0,1]}, \phi^2_{1,0}, \phi^3_{1,0}, \phi^4_{1,0}, \phi^1_{1,1}, \phi^2_{1,1}, \phi^3_{1,1}, \phi^4_{1,1}, \phi^1_{1,2}\chi_{[0,1]}\right\} \tag{4.77}$$

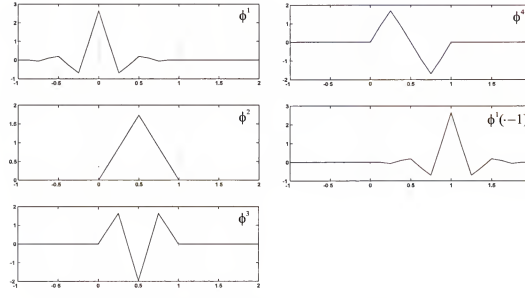


Figure 4.15: Level 0 Scaling Functions That Intersect [0,1]

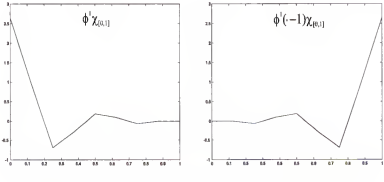


Figure 4.16: Boundary-Adapted Scaling Functions

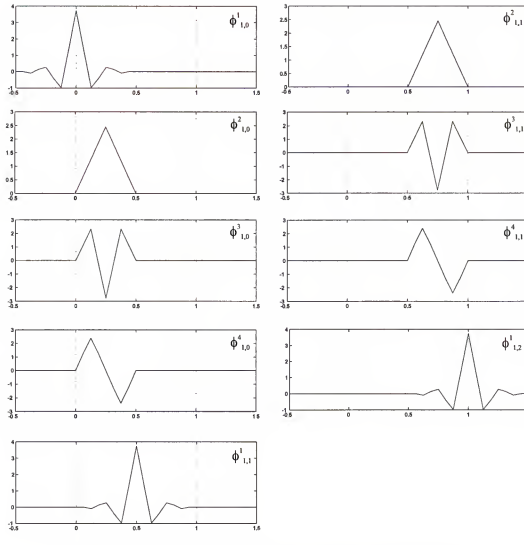


Figure 4.17: Level 1 Scaling Functions That Intersect [0,1]

In general, the space $V_j[0,1]$ has dimension $4 \cdot 2^j + 1$ and is defined as

$$V_j[0,1] := span\left\{\phi^1_{j,0}\chi_{[0,1]}, \phi^1_{j,k}|_{k=1}^{2^j-1}, \phi^2_{j,k}|_{k=0}^{2^j-1}, \phi^3_{j,k}|_{k=0}^{2^j-1}, \phi^4_{j,k}|_{k=0}^{2^j-1}, \phi^1_{j,2^j}\chi_{[0,1]}\right\} \quad (4.78)$$

In constructing the spaces $\{V_j[0,1]\}$ that form a multiresolution analysis for $L^2[0,1]$, every scaling function that intersects $[0,1]$ is included in the basis. If the function overlaps the boundary, it is simply restricted to $[0,1]$ and included in the basis. This simple procedure cannot be used for the wavelet spaces $\{W_j[0,1]\}$, however. To demonstrate this, consider the space $W_0[0,1]$, defined as

$$W_0[0,1] := V_1[0,1] \ominus V_0[0,1] \tag{4.79}$$

This space is constructed using the four piecewise-linear wavelets $\{\psi^1, \psi^2, \psi^3, \psi^4\}$. The wavelets ψ^1 and ψ^2 are supported over $[-1,1]$ while the wavelets ψ^3 and ψ^4 are both supported over $[0,1]$. As shown in Figure (4.18), there are a total of six wavelets that intersect the domain $[0,1]$. The wavelets $\{\psi^1, \psi^2, \psi^1(\cdot - 1), \psi^2(\cdot - 1)\}$ all overlap the boundaries. Recall that the dimension of the space $V_0[0,1]$ is five while the dimension of $V_1[0,1]$ is nine. Then, equation (4.79) implies that

$$dim\,(W_0[0,1]) = dim\,(V_1[0,1]) - dim\,(V_0[0,1]) = 4$$

Therefore, the space $W_0[0,1]$ cannot simply be defined as the span of the six wavelets $\{\psi^1\chi_{[0,1]}, \psi^2\chi_{[0,1]}, \psi^3, \psi^4, \psi^1(\cdot - 1)\chi_{[0,1]}, \psi^2(\cdot - 1)\chi_{[0,1]}\}$. Indeed, equation (4.79) implies that these wavelets do not form a linearly independent set because two of them are redundant. In determining which two functions to exclude from the basis for $W_0[0,1]$, recall that the spaces $V_0[0,1]$ and $W_0[0,1]$ are required to be orthogonal. The anti-symmetric wavelet ψ^2 is automatically orthogonal to the symmetric scaling function ϕ^1. This does not imply, however, that the individual halves of these functions are orthogonal. That is, in general,

$$\left\langle \psi^2\chi_{[0,1]}, \phi^1\chi_{[0,1]} \right\rangle \neq 0$$
$$\left\langle \psi^2(\cdot - 1)\chi_{[0,1]}, \phi^1(\cdot - 1)\chi_{[0,1]} \right\rangle \neq 0$$

Therefore, $\psi^2\chi_{[0,1]}$ and $\psi^2(\cdot - 1)\chi_{[0,1]}$ should not be included in the $W_0[0,1]$ basis. On the other hand, by construction, since the functions ψ^1 and ϕ^1 are orthogonal, and both are symmetric over $[-1,1]$, they must satisfy

$$\langle \psi^1 \chi_{[0,1]}, \phi^1 \chi_{[0,1]} \rangle = 0$$
$$\langle \psi^1(\cdot - 1)\chi_{[0,1]}, \phi^1(\cdot - 1)\chi_{[0,1]} \rangle = 0$$

Therefore, the wavelets $\{\psi^1, \psi^3, \psi^4, \psi^1(\cdot - 1)\}$ form an orthogonal set and are orthogonal to the scaling functions that span $V_0[0,1]$. The space $W_0[0,1]$ is then defined as

$$W_0[0,1] := span\left\{\psi^1 \chi_{[0,1]}, \psi^3, \psi^4, \psi^1(\cdot - 1)\chi_{[0,1]}\right\} \tag{4.80}$$

It is important to note that the antisymmetric wavelet ψ^2 is not excluded from the finer-resolution wavelet spaces. For example, the space $W_1[0,1]$ is defined as

$$W_1[0,1] := span\left\{\psi^1_{1,0}\chi_{[0,1]}, \psi^3_{1,0}, \psi^4_{1,0}, \psi^1_{1,1}, \psi^2_{1,1}, \psi^3_{1,1}, \psi^4_{1,1}, \psi^1_{1,2}\chi_{[0,1]}\right\} \tag{4.81}$$

where, as usual,

$$\psi^s_{j,k} = 2^{j/2}\psi^s(2^j - k) \tag{4.82}$$

The wavelets $\psi^2_{1,0}\chi_{[0,1]}$ and $\psi^2_{1,2}\chi_{[0,1]}$ intersect the boundary and are excluded from the $W_1[0,1]$ basis, but the "interior" wavelet $\psi^2_{1,1}$ is needed in the basis. In general, the space $W_j[0,1]$ has dimension $4 \cdot 2^j$ and is defined as

$$W_j[0,1] := span\left\{\psi^1_{j,0}\chi_{[0,1]}, \psi^1_{j,k}|_{k=1}^{2^j-1}, \psi^2_{j,k}|_{k=1}^{2^j-1}, \psi^3_{j,k}|_{k=0}^{2^j-1}, \psi^4_{j,k}|_{k=0}^{2^j-1}, \psi^1_{j,2^j}\chi_{[0,1]}\right\} \tag{4.83}$$

The extension of the foregoing results to higher dimensions is relatively straightforward. In higher dimensions, the scaling functions and wavelets are formed as tensor products of the one-dimensional, boundary-adapted functions using the same

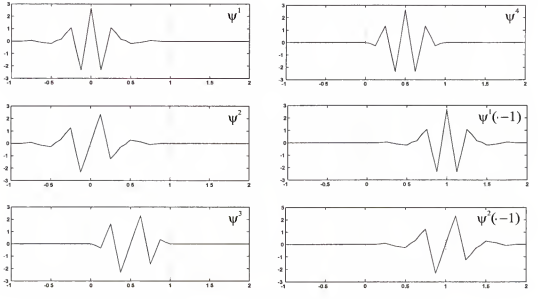


Figure 4.18: Level 0 Wavelets That Intersect [0,1]

procedure that was described in Section 3.6. Defining Γ as the $[0,1]$ x $[0,1]$ square domain, the coarsest-scale space $V_0(\Gamma)$ is spanned by a total of 25 two-dimensional linear scaling functions, obtained as the tensor products of the five one-dimensional scaling functions in the first dimension (x-direction) with those in the second dimension (y-direction). Similarly, the wavelet spaces $W_0^1(\Gamma)$ and $W_0^2(\Gamma)$ are each spanned by a total of 20 two-dimensional wavelets. These are obtained as the tensor products of the five scaling functions in the x-direction with the four wavelets in the y-direction for $W_0^1(\Gamma)$ and the four wavelets in the x-direction with the five scaling functions in the y-direction for $W_0^2(\Gamma)$. Finally, the wavelet space $W_0^3(\Gamma)$ is spanned by 16 wavelets formed as the tensor products of the four wavelets in the x-direction with those in the y-direction. In general, the space $V_j(\Gamma)$ is spanned by $(4 \cdot 2^j + 1)^2$ two-dimensional scaling functions while the wavelet spaces $W_j^1(\Gamma)$, $W_j^2(\Gamma)$, and $W_j^3(\Gamma)$ are spanned by $(4 \cdot 2^j + 1)(4 \cdot 2^j)$, $(4 \cdot 2^j)(4 \cdot 2^j + 1)$, and $(4 \cdot 2^j)^2$ wavelets, respectively. The generalization to three dimensions and higher follows in the same manner.

As a final note, although the above example specifically uses the piecewise-linear multiwavelets, the procedure for deriving boundary-adapted scaling functions and

wavelets directly generalizes to the higher-order multiwavelets. This is because, regardless of the order of the scaling functions used, there is always exactly one symmetric scaling function supported over $[-1, 1]$ while all the others are supported over $[0, 1]$. Similarly, there is always exactly one symmetric and one antisymmetric wavelet supported over $[-1, 1]$ while the others are all supported over $[0, 1]$. Therefore, regardless of the order of the multiwavelets, there are always the same number of functions that overlap the boundary, and the procedure for dealing with these functions does not change. The only difference is that as the order is increased, there are additional functions supported over $[0, 1]$ and consequently more total functions.

4.7 Multiwavelet Decomposition Example

To complete the discussion of multiwavelets, the piecewise-linear, quadratic, and cubic multiwavelets are used to decompose the function shown in Figure (4.19). Recall that the Haar wavelet transform was demonstrated on this same function in Section 3.3. The function is a .5 Hz sine wave over the domain $[0, 1]$ with a sharp feature in the form of a 32 Hz sine component. Because the function is supported on a finite domain, the method described in the previous section is used to adapt the multiwavelets to the boundaries. The decomposition of this function using the piecewise-linear, quadratic, and cubic multiwavelets is shown in Figures (4.20) through (4.22). In all three cases, the original fine-scale representation of the function is taken at level 8 and the coarsest level is chosen to be level 0. Therefore, the multiscale wavelet representation of the function is given in terms of the scaling function approximation on level 0 and the wavelet details on levels 0 through 7. Note that, on the finer-resolution levels, the wavelet details are nonzero only where the sharp features of the function occur. On the coarser scales, there are no details from these sharp features and the wavelet details focus on the low frequency component of the signal. The effect of the approximation order of the multiwavelets is also evident from the figures. As the approximation

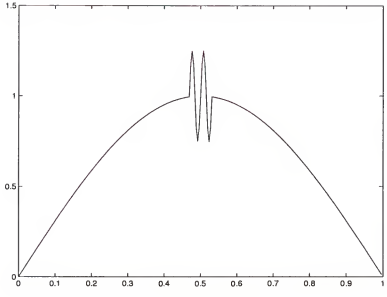


Figure 4.19: Original Function

order increases, the scaling functions yield better approximations of the function at any given level.

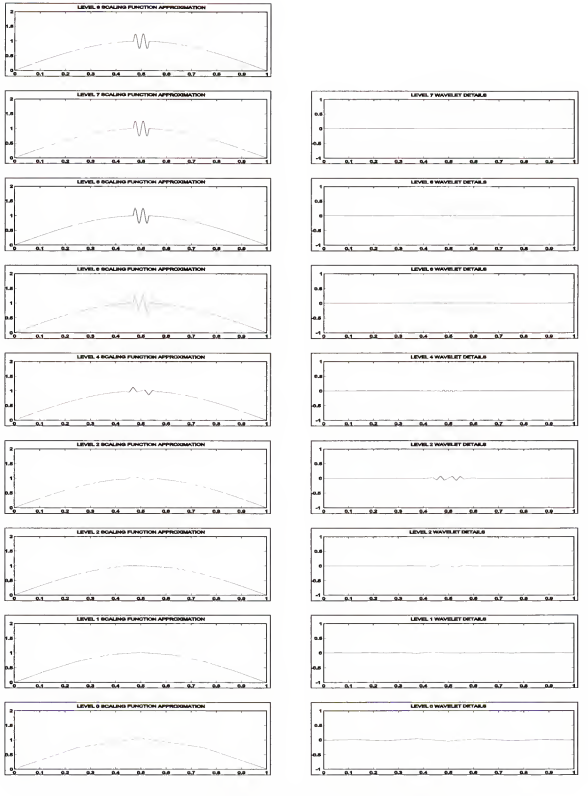


Figure 4.20: Piecewise-Linear Multiwavelet Decomposition

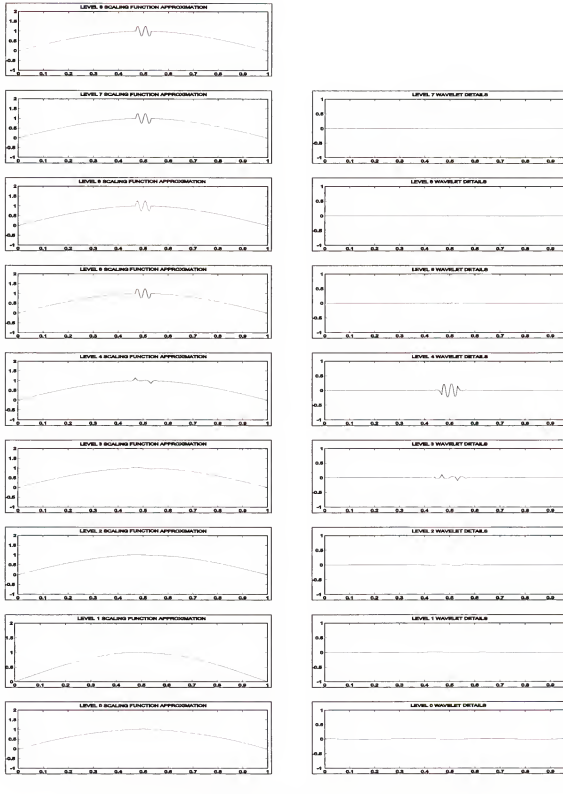


Figure 4.21: Piecewise-Quadratic Multiwavelet Decomposition

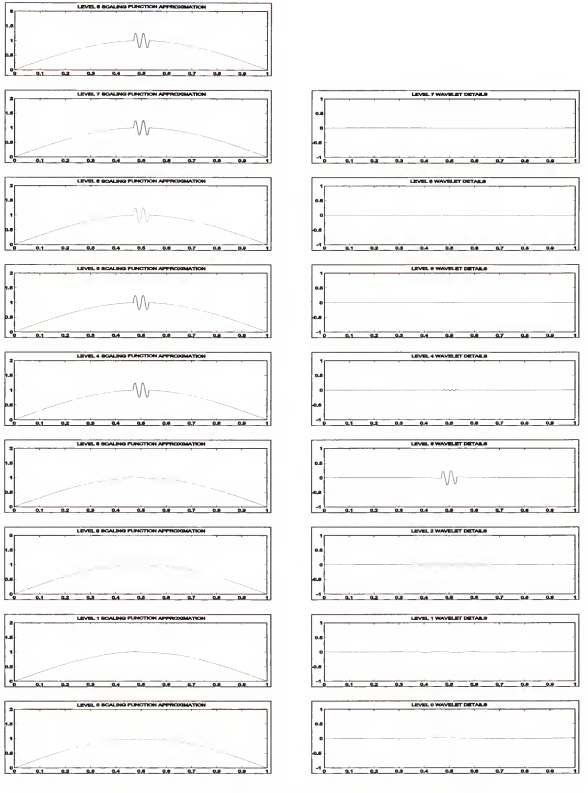


Figure 4.22: Piecewise-Cubic Multiwavelet Decomposition

CHAPTER 5
WAVELETS SUPPORTED ON TRIANGULAR DOMAINS

In Chapter 2, it was demonstrated that when the dependent variable of a time-invariant, nonlinear ODE is expressed in terms of a Volterra series, the second-order Volterra operator takes the form

$$y_2(t) = \int_0^t \int_0^\eta h_2(\xi, \eta)\, u(t-\xi)\, u(t-\eta)\, d\xi d\eta \tag{5.1}$$

Equation (5.1) shows that the second-order operator is dependent on the second-order Volterra kernel h_2 over the triangular domain defined by $0 \leq \xi \leq \eta$ and $0 \leq \eta \leq t$. Therefore, in many cases, it is natural and convenient to use the triangular form of the kernel instead of the symmetric form over the $[0, t]$ x $[0, t]$ square domain. It is assumed that the kernel is a decaying function and that, for some time D, the kernel is essentially zero for $t > D$. This implies that the output of the system is not influenced by inputs occurring further back in time than D (i.e., the system has finite memory). Then, the second-order kernel is supported over the triangular domain Ω shown in Figure (5.1).

5.1 Construction of Triangular Wavelets

In order to develop wavelet-based representations of the triangular second-order Volterra kernel, it is necessary to derive a multiresolution analysis over the domain Ω. The construction of wavelets that are supported on compact subsets of $\mathbb{R}^d$, where d is any positive integer, has been addressed in a series of papers by Micchelli, Xu, and Chen [4], [5], [101]. In Ref. [4], a procedure is outlined whereby a multiresolution

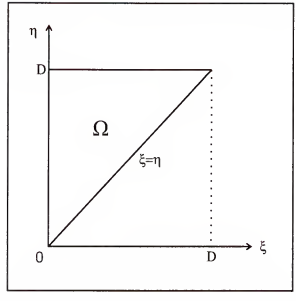


Figure 5.1: Triangular Domain of Support for the Second-Order Volterra Kernel

analysis can be constructed for a compact set $E \subset \mathbb{R}^d$ for which there exists a family of m contractive affine mappings

$$\gamma_i : \mathbb{R}^d \longrightarrow \mathbb{R}^d, \qquad i = 0, 1, ..., m-1$$

such that

$$E = \bigcup_{i=0}^{m-1} \gamma_i(E) \tag{5.2}$$

and

$$meas\left(\gamma_i(E) \bigcap \gamma_j(E)\right) = 0, \quad i \neq j \tag{5.3}$$

where $meas(A)$ denotes the Lebesque measure of a set $A \subseteq \mathbb{R}^d$. A set E that satisfies these conditions is termed an invariant set relative to the family of mappings $\Gamma := \{\gamma_i : \ i = 0, ..., m-1\}$.

The method detailed in Ref. [4] can be applied to the construction of a multiresolution analysis over the triangular domain $\Omega \subset \mathbb{R}^2$. First, we define four contractive affine mappings

$$\gamma_i : \mathbb{R}^2 \longrightarrow \mathbb{R}^2, \qquad i = 0, \ldots, 3$$

which act on any point $(\xi, \eta) \in \mathbb{R}^2$ as follows:

$$\gamma_0(\xi,\eta) = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{Bmatrix} \xi \\ \eta \end{Bmatrix} \tag{5.4}$$

$$\gamma_1(\xi,\eta) = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{Bmatrix} \xi \\ \eta \end{Bmatrix} + \begin{Bmatrix} \frac{D}{2} \\ \frac{D}{2} \end{Bmatrix} \tag{5.5}$$

$$\gamma_2(\xi,\eta) = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{Bmatrix} \xi \\ \eta \end{Bmatrix} + \begin{Bmatrix} 0 \\ \frac{D}{2} \end{Bmatrix} \tag{5.6}$$

$$\gamma_3(\xi,\eta) = \begin{bmatrix} -\frac{1}{2} & 0 \\ 0 & -\frac{1}{2} \end{bmatrix} \begin{Bmatrix} \xi \\ \eta \end{Bmatrix} + \begin{Bmatrix} \frac{D}{2} \\ D \end{Bmatrix} \tag{5.7}$$

These mappings transform the set $\Omega \subset \mathbb{R}^2$ into the four subsets

$$\Omega_i := \gamma_i(\Omega), \qquad i = 0, \ldots, 3$$

shown in Figure (5.2). Clearly,

$$\Omega = \bigcup_{i=0}^{3} \Omega_i$$

and the subsets Ω_i do not intersect,

$$meas\left(\Omega_i \bigcap \Omega_j\right) = 0, \qquad i \neq j$$

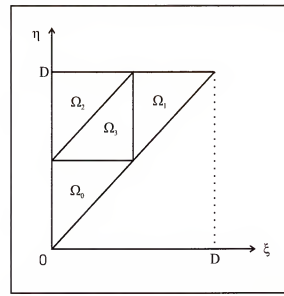


Figure 5.2: Subsets $\Omega_i \subset \Omega$

Then, it follows that Ω is an invariant subset of $\mathbb{R}^2$ relative to the family of mappings $\Gamma := \{\gamma_i : \quad i = 0, \ldots, 3\}$. The mappings in Γ are invertible and it can easily be verified that the inverse mappings

$$\gamma_i^{-1} : \mathbb{R}^2 \longrightarrow \mathbb{R}^2, \qquad i = 0, \ldots, 3$$

are given by

$$\gamma_0^{-1}(\xi, \eta) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{Bmatrix} \xi \\ \eta \end{Bmatrix} \tag{5.8}$$

$$\gamma_1^{-1}(\xi, \eta) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{Bmatrix} \xi \\ \eta \end{Bmatrix} - \begin{Bmatrix} D \\ D \end{Bmatrix} \tag{5.9}$$

$$\gamma_2^{-1}(\xi, \eta) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{Bmatrix} \xi \\ \eta \end{Bmatrix} - \begin{Bmatrix} 0 \\ D \end{Bmatrix} \tag{5.10}$$

$$\gamma_3^{-1}(\xi, \eta) = \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix} \begin{Bmatrix} \xi \\ \eta \end{Bmatrix} - \begin{Bmatrix} -D \\ -2D \end{Bmatrix} \tag{5.11}$$

The first step towards generating a multiresolution analysis over Ω is to find an n-dimensional refinable curve $\underline{f} \in L^2(\Omega)$. This curve defines a subspace $V_0 \subset L^2(\Omega)$:

$$V_0 := \{\underline{c}^T \underline{f} : \quad \underline{c} \in \mathbb{R}^n\} \tag{5.12}$$

The components f_i, $i = 1, ..., n$, represent basis functions for the subspace V_0. For a given set E, a vector field $\underline{f}(t)$, $t \in E$, is refinable if there exist n x n matrices $[A_0,], ..., [A_{m-1}]$ such that

$$G_i \underline{f}(t) = [A_i]^T \underline{f}(t), \qquad i = 0, ..., m-1 \tag{5.13}$$

and

$$\underline{v}^T \underline{f}(t) = a \tag{5.14}$$

where $\underline{v} \in \mathbb{R}^n$ and a is a nonzero constant [5]. In equation (5.13), G_i , $i = 0, \ldots, m-1$, represent operators defined as

$$G_i f(t) := f \circ \gamma_i(t) = f(\gamma_i(t)) \tag{5.15}$$

so that

$$G_i \underline{f}(t) = \begin{Bmatrix} G_i f_1(t) \\ \vdots \\ G_i f_n(t) \end{Bmatrix} = \begin{Bmatrix} f_1 \circ \gamma_i(t) \\ \vdots \\ f_n \circ \gamma_i(t) \end{Bmatrix}, \qquad i = 0, \ldots, m-1$$

A simple choice of a refinable curve in $L^2(\Omega)$ is the one-dimensional vector $f = \chi_\Omega$, where χ_Ω is the characteristic function of the set Ω, defined as

$$\chi_\Omega(\xi, \eta) := \begin{cases} 1 & (\xi, \eta) \in \Omega \\ 0 & \text{otherwise} \end{cases} \tag{5.16}$$

To show that χ_Ω is a refinable curve in $L^2(\Omega)$, it must be demonstrated that equations (5.13) and (5.14) are satisfied for any point $(\xi, \eta) \in \Omega$. In this case, equations (5.13) and (5.14) take the form

$$G_i\chi_\Omega(\xi,\eta) = A_i\chi_\Omega(\xi,\eta), \quad i = 0,\ldots,3 \tag{5.17}$$

$$v\chi_\Omega(\xi,\eta) = a \tag{5.18}$$

where A_i, $i = 0,\ldots,3$, and v are now nonzero constants. Using equation (5.15),

$$G_i\chi_\Omega(\xi,\eta) = \chi_\Omega \circ \gamma_i(\xi,\eta) = \chi_\Omega\left(\gamma_i(\xi,\eta)\right), \quad i = 0,\ldots,3$$

Clearly, any point $(\xi,\eta) \in \Omega$ subject to the mapping γ_i will be mapped into the set Ω_i. Since $\Omega_i \subset \Omega$ for $i = 0,1,2,3$, and using the definition of the characteristic function χ_Ω given in equation (5.16), it follows that

$$G_i\chi_\Omega(\xi,\eta) = \chi_\Omega\left(\gamma_i(\xi,\eta)\right) = 1, \quad i = 0,\ldots,3$$

for all $(\xi,\eta) \in \Omega$. Then, equation (5.17) is satisfied by choosing $A_i = 1$, $i = 0,\ldots,3$. Equation (5.18) is trivially satisfied by choosing $v = 1$. Then, $a = 1$ for all $(\xi,\eta) \in \Omega$.

Now that it has been established that χ_Ω represents a refinable curve in $L^2(\Omega)$, the subspace $V_0 \subset L^2(\Omega)$ is defined as

$$V_0 := \{c_1\chi_\Omega : \quad c_1 \in \mathbb{R}\} \tag{5.19}$$

That is, V_0 is the one-dimensional vector space spanned by the characteristic function χ_Ω. In Ref. [4], it is shown that, given the coarsest-scale space V_0, a sequence of nested spaces

$$V_{j+1} \supseteq V_j, \quad j = 0,1,2,\ldots$$

can be constructed. This result is achieved via the following family of linear operators on $L^2(\Omega)$:

$$(T_i g)(\xi,\eta) = \sum_{j=0}^{3} q_{ij}\, g\left(\gamma_j^{-1}(\xi,\eta)\right) \chi_{\Omega_j}(\xi,\eta), \qquad i = 0,\ldots,3 \tag{5.20}$$

In equation (5.20), g represents any function in $L^2(\Omega)$ and χ_{Ω_j} denotes the characteristic function over the set Ω_j. The $\{q_{ij}\}_{i,j=0}^{3}$ are the entries of a 4 x 4 orthogonal matrix $[Q]$. Because $[Q]$ is orthogonal, it satisfies the condition

$$[Q]^T[Q] = [I] \tag{5.21}$$

where $[I]$ is the 4 x 4 identity matrix. Using the operators $\{T_i\}$ in equation (5.20), the finer-resolution spaces $\{V_{j+1}\}$ are constructed from the coarser-resolution spaces $\{V_j\}$ via the recurrence formula

$$V_{j+1} := \bigoplus_{i=0}^{3} T_i V_j, \qquad j = 0,1,2,\ldots \tag{5.22}$$

where the symbol $\bigoplus$ denotes the sum of spaces. The orthogonality of $[Q]$ ensures that the spaces $\{T_i V_j\}$ do not intersect:

$$\bigcap_{i=0}^{3} T_i V_j = 0$$

In applying equation (5.22), we first choose the matrix $[Q]$ to be the 4 x 4 identity matrix. Then, the operators in equation (5.20) take the simplified form

$$(T_i g)(\xi,\eta) = g\left(\gamma_i^{-1}(\xi,\eta)\right) \chi_{\Omega_i}(\xi,\eta), \qquad i = 0,\ldots,3 \tag{5.23}$$

In simple terms, the operators $\{T_i\}$ restrict functions in Ω to the subsets $\{\Omega_i\} \subset \Omega$ in the same manner as the mappings $\{\gamma_i\}$ map points in Ω into $\{\Omega_i\}$. Equation (5.22) can now be used to construct the subspace V_1 from V_0:

$$V_1 = T_0 V_0 \oplus T_1 V_0 \oplus T_2 V_0 \oplus T_3 V_0 \tag{5.24}$$

Recall that $V_0 = span\{\chi_\Omega\}$. The operator T_0 in equation (5.23) acts on χ_Ω in the following manner:

$$(T_0\chi_\Omega)(\xi,\eta) = \chi_\Omega(\gamma_0^{-1}(\xi,\eta))\chi_{\Omega_0}(\xi,\eta) \tag{5.25}$$

The function χ_{Ω_0} in equation (5.25) requires that $(\xi,\eta) \in \Omega_0$ in order for $(T_0\chi_\Omega)(\xi,\eta)$ to be nonzero. The operator γ_0^{-1} maps any $(\xi,\eta) \in \Omega_0$ into the larger set Ω. Then,

$$\chi_\Omega\left(\gamma_0^{-1}(\xi,\eta)\right) = 1 \qquad \forall\ (\xi,\eta) \in \Omega_0$$

Thus, we have

$$(T_0\chi_\Omega)(\xi,\eta) = \chi_{\Omega_0}(\xi,\eta) \tag{5.26}$$

In a similar manner, we obtain

$$(T_1\chi_\Omega)(\xi,\eta) = \chi_{\Omega_1}(\xi,\eta) \tag{5.27}$$

$$(T_2\chi_\Omega)(\xi,\eta) = \chi_{\Omega_2}(\xi,\eta) \tag{5.28}$$

$$(T_3\chi_\Omega)(\xi,\eta) = \chi_{\Omega_3}(\xi,\eta) \tag{5.29}$$

Then, the space V_1 is defined as the vector space spanned by the characteristic functions of the sets $\{\Omega_i\}$ shown in Figure (5.2):

$$V_1 := span\{\chi_{\Omega_i} : \quad i = 0,\ldots,3\} \tag{5.30}$$

It should be noted that $dim(V_1) = 4$ and that, in general, $dim(V_j) = 4^j$. Noting that

$$\chi_\Omega(\xi,\eta) = \sum_{i=0}^{3} \chi_{\Omega_i}(\xi,\eta)$$

it is clear that $V_0 \subset V_1$ since any function in V_0 can also be expressed in terms of the basis functions of V_1.

Now that the space V_1 has been generated from V_0, equation (5.22) can be used again to construct the space V_2 from V_1:

$$V_2 = T_0V_1 \oplus T_1V_1 \oplus T_2V_1 \oplus T_3V_1 \tag{5.31}$$

The basis functions for the space T_0V_1 are obtained by applying the operator T_0 to the basis functions of V_1. For example,

$$(T_0\chi_{\Omega_0})(\xi,\eta) = \chi_{\Omega_0}\left(\gamma_0^{-1}(\xi,\eta)\right)\chi_{\Omega_0}(\xi,\eta)$$

It is not difficult to verify that $(T_0\chi_{\Omega_0})(\xi,\eta)$ is equal to one if $(\xi,\eta) \in \Omega_{00}$ and zero otherwise, where Ω_{00} is the subset of Ω shown in Figure (5.3). Therefore,

$$(T_0\chi_{\Omega_0})(\xi,\eta) = \chi_{\Omega_{00}}(\xi,\eta) \tag{5.32}$$

where $\chi_{\Omega_{00}}(\xi,\eta)$ is the characteristic function of the set Ω_{00}. Following the same argument, in general we have

$$(T_i\chi_{\Omega_j})(\xi,\eta) = \chi_{\Omega_j}(\gamma_i^{-1}(\xi,\eta))\chi_{\Omega_i}(\xi,\eta) = \chi_{\Omega_{ij}}(\xi,\eta), \quad i,j \in \{0,1,2,3\} \tag{5.33}$$

Therefore, V_2 is the 16-dimensional subspace spanned by the characteristic functions over the sets Ω_{ij} shown in Figure (5.3). That is,

$$V_2 = span\left\{\chi_{\Omega_{ij}} : \ i,j \in \{0,1,2,3\}\right\} \tag{5.34}$$

By inspection, it is clear that $V_1 \subset V_2$. Equation (5.22) can be applied recursively to obtain spaces $\{V_j\} \subset L^2(\Omega)$ of increasingly finer resolution. In Ref. [4], it is proven that

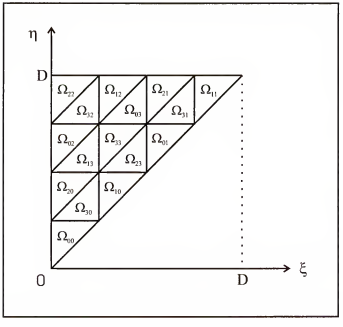


Figure 5.3: Subsets $\Omega_{ij} \subset \Omega$

$$\overline{\bigcup_{j=0}^{\infty} V_j} = L^2(\Omega) \tag{5.35}$$

meaning that as j approaches infinity, the spaces $\{V_j\}$ approach $L^2(\Omega)$.

To this point, a series of nested subspaces $\{V_j\} \subset L^2(\Omega)$ has been generated using the technique of Micchelli and Xu [4]. The basis functions for these spaces are the scaling functions, or generators, of the multiresolution analysis. It now remains to construct the orthogonal complement spaces $\{W_j\}$ such that

$$W_j = V_{j+1} \ominus V_j, \qquad j = 0, 1, 2, \ldots \tag{5.36}$$

where the spaces V_j and W_j are orthogonal. Before doing this, it is convenient to normalize the scaling functions that form the bases for the various spaces $\{V_j\}$. Working with orthonormal bases simplifies the decomposition and reconstruction formulas that will be derived later. Towards this end, we start with the subspace V_0, which is the one-dimensional space spanned by the characteristic function over Ω, χ_Ω. Denoting A_0 to be the area of the triangular domain Ω, it easy to verify that the function

$$\phi(\xi,\eta) := \frac{1}{\sqrt{A_0}}\chi_\Omega(\xi,\eta) \tag{5.37}$$

is an orthonormal basis for V_0:

$$\langle\phi,\phi\rangle_{L^2(\Omega)} = \int_\Omega \phi^2(\xi,\eta)\,d\Omega = \frac{1}{A_0}\int_\Omega d\Omega = 1$$

From this point forward, the subspace V_0 will be defined as

$$V_0 := span\{\phi\} \tag{5.38}$$

To derive orthonormal scaling functions for the other spaces $\{V_j\}$, $j \geq 1$, we first note that the basis functions for these subspaces are already orthogonal since they are composed of characteristic functions that do not intersect. Following the same argument as for the basis function of V_0, the basis functions of each space V_j will be normalized if they are scaled by a factor of $A_j^{-1/2}$, where A_j is the area of the domain over which the functions are supported. At this point, it is convenient to introduce some new notation for the scaling functions that span the spaces $\{V_j\}$. Each scaling function can be parameterized by two indices, j and $\mathbb{K}$, where j represents the resolution level of the function and $\mathbb{K}$ represents a j-dimensional multi-index $\mathbb{K} = k_j k_{j-1}...k_1$, where $k_i \in \{0,1,2,3\}$, $i = 1,...,j$. The multi-index $\mathbb{K}$ gives the sequence of operators T_i that transforms the characteristic function over Ω into the given function. For example, the function $\chi_{\Omega_{01}}$ is obtained as

$$\chi_{\Omega_{01}}(\xi,\eta) = T_1 \circ \chi_{\Omega_0}(\xi,\eta) = T_1 \circ T_0 \circ \chi_\Omega(\xi,\eta)$$

Therefore, $\chi_{\Omega_{01}}$ is parameterized by the indices $j = 2$ and $\mathbb{K} = 01$ and can be written as $\chi_{2,01}$. In general, noting that

$$A_j = 2^{-2j}A_0 \Longrightarrow \frac{1}{\sqrt{A_j}} = 2^j\frac{1}{\sqrt{A_0}}$$

the orthonormal scaling functions are defined as

$$\phi_{j,\mathbb{K}}(\xi,\eta) := \frac{1}{\sqrt{A_j}}\chi_{j,\mathbb{K}}(\xi,\eta) = 2^j(T_{k_j} \circ T_{k_{j-1}} \circ \cdots \circ T_{k_1} \circ \phi(\xi,\eta)) \tag{5.39}$$

For convenience, Λ_j is defined as the set of all possible multi-indices of length j, where each member of the multi-index is an element of the set $\{0,1,2,3\}$. Then, the spaces $\{V_j\}$ are defined as

$$V_j := span\,\{\phi_{j,\mathbb{K}}(\xi,\eta)\}_{\mathbb{K}\in\Lambda_j} \tag{5.40}$$

Now that we have generated orthonormal scaling functions that span the spaces $\{V_j\}$, the wavelets that span the orthogonal complement spaces $\{W_j\}$ can be constructed. Micchelli and Xu [4] have shown that, once a basis for the coarsest-scale wavelet space W_0 is obtained, bases for the finer-resolution spaces $\{W_j\}$ can be generated through a recursive application of the operators $\{T_i\}$. With this in mind, a wavelet basis for the orthogonal complement space W_0 that satisfies

$$W_0 = V_1 \ominus V_0 \tag{5.41}$$

where the basis functions of W_0 are orthogonal to the basis function of V_0, must be constructed. Recall that

$$V_1 = span\,\{\phi_{1,0}, \phi_{1,1}, \phi_{1,2}, \phi_{1,3}\}$$

where the scaling functions are defined as in equation (5.39). Since $dim(V_1) = 4$ and $dim(V_0) = 1$, we must have $dim(W_0) = 3$. Therefore, three wavelets $\{\psi^1, \psi^2, \psi^3\}$ must be constructed that are orthogonal to the scaling function ϕ that spans V_0 and satisfy equation (5.41). In addition, the wavelets will be constructed to be mutually orthonormal so that they satisfy

$$\langle \psi^r, \psi^s \rangle_{L^2(\Omega)} = \int_{\Omega} \psi^r(\xi,\eta)\psi^s(\xi,\eta)d\Omega = \delta_{r,s} \tag{5.42}$$

and form an orthonormal basis for W_0. In order to ensure that equation (5.41) is satisfied, the three wavelets must be linear combinations of the four scaling functions that span V_1:

$$\psi^r(\xi,\eta) = \sum_{i=0}^{3} c_i^r \phi_{1,i}(\xi,\eta), \qquad r = 1,2,3 \tag{5.43}$$

It is easy to verify by inspection that

$$\phi(\xi,\eta) = \frac{1}{2}\phi_{1,0}(\xi,\eta) + \frac{1}{2}\phi_{1,1}(\xi,\eta) + \frac{1}{2}\phi_{1,2}(\xi,\eta) + \frac{1}{2}\phi_{1,3}(\xi,\eta) \tag{5.44}$$

Equations (5.43) and (5.44) can be combined into the following matrix form:

$$\begin{Bmatrix} \phi \\ \psi^1 \\ \psi^2 \\ \psi^3 \end{Bmatrix} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ c_0^1 & c_1^1 & c_2^1 & c_3^1 \\ c_0^2 & c_1^2 & c_2^2 & c_3^2 \\ c_0^3 & c_1^3 & c_2^3 & c_3^3 \end{bmatrix} \begin{Bmatrix} \phi_{1,0} \\ \phi_{1,1} \\ \phi_{1,2} \\ \phi_{1,3} \end{Bmatrix} \tag{5.45}$$

The coefficients $\{c_i^r\}$ must be chosen in such a manner that the coefficient matrix in equation (5.45) is of full rank. Then, the functions $\{\phi, \psi^1, \psi^2, \psi^3\}$ form a linearly independent set. If the coefficients are chosen such that

$$\begin{aligned} \{c_0^1, c_1^1, c_2^1, c_3^1\} &= \{\alpha, -\alpha, 0, 0\} \\ \{c_0^2, c_1^2, c_2^2, c_3^2\} &= \{0, 0, \beta, -\beta\} \\ \{c_0^3, c_1^3, c_2^3, c_3^3\} &= \{\gamma, \gamma, -\gamma, -\gamma\} \end{aligned}$$

the coefficient matrix takes the form

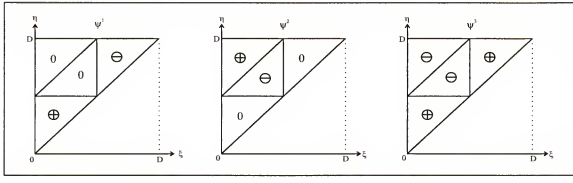


Figure 5.4: Wavelets That Form a Basis for W_0

$$\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \alpha & -\alpha & 0 & 0 \\ 0 & 0 & \beta & -\beta \\ \lambda & \lambda & -\lambda & -\lambda \end{bmatrix} \tag{5.46}$$

Clearly, the coefficient matrix shown in equation (5.46) is of full rank, regardless of the values of α, β, and λ. The wavelets then take the form shown in Figure (5.4), where the regions are marked $\oplus$, $\ominus$, or 0 to show where the functions are positive, negative, or zero. By inspection, it is clear that the wavelets are all orthogonal to the scaling function ϕ, which is a positive constant over the entire domain Ω. Therefore, these wavelets form a basis for the orthogonal complement space W_0. Referring again to Figure (5.4), it is apparent that the wavelets are mutually orthogonal. The only remaining task is to calculate values for α, β, and λ such that the wavelets are orthonormal. The following equations must be satisfied:

$$\int_{\Omega} \left[\psi^1(\xi,\eta)\right]^2 d\Omega = \alpha^2 \int_{\Omega_0} \left[\phi_{1,0}(\xi,\eta)\right]^2 d\Omega_0 + \alpha^2 \int_{\Omega_1} \left[\phi_{1,1}(\xi,\eta)\right]^2 d\Omega_1 = 1 \tag{5.47}$$

$$\int_{\Omega} \left[\psi^2(\xi,\eta)\right]^2 d\Omega = \beta^2 \int_{\Omega_2} \left[\phi_{1,2}(\xi,\eta)\right]^2 d\Omega_2 + \beta^2 \int_{\Omega_3} \left[\phi_{1,3}(\xi,\eta)\right]^2 d\Omega_3 = 1 \tag{5.48}$$

$$\int_{\Omega} \left[\psi^3(\xi,\eta)\right]^2 d\Omega = \lambda^2 \int_{\Omega_0} \left[\phi_{1,0}(\xi,\eta)\right]^2 d\Omega_0 + \lambda^2 \int_{\Omega_1} \left[\phi_{1,1}(\xi,\eta)\right]^2 d\Omega_1 \tag{5.49}$$

$$+\lambda^2 \int_{\Omega_2} [\phi_{1,2}(\xi,\eta)]^2 d\Omega_2 + \lambda^2 \int_{\Omega_3} [\phi_{1,3}(\xi,\eta)]^2 d\Omega_3 = 1$$

Recall that the scaling functions are orthonormal. Then, the integrals of the form

$$\int_{\Omega_i} [\phi_{1,i}(\xi,\eta)]^2 d\Omega_i, \qquad i = 0,\ldots,3$$

reduce to 1 and we have

$$\{\alpha,\beta,\lambda\} = \left\{\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, \frac{1}{2}\right\}$$

Therefore, the orthonormal wavelets are defined as

$$\psi^1(\xi,\eta) := \frac{1}{\sqrt{2}}\phi_{1,0}(\xi,\eta) - \frac{1}{\sqrt{2}}\phi_{1,1}(\xi,\eta) \tag{5.50}$$

$$\psi^2(\xi,\eta) := \frac{1}{\sqrt{2}}\phi_{1,2}(\xi,\eta) - \frac{1}{\sqrt{2}}\phi_{1,3}(\xi,\eta) \tag{5.51}$$

$$\psi^3(\xi,\eta) := \frac{1}{2}\phi_{1,0}(\xi,\eta) + \frac{1}{2}\phi_{1,1}(\xi,\eta) - \frac{1}{2}\phi_{1,2}(\xi,\eta) - \frac{1}{2}\phi_{1,3}(\xi,\eta) \tag{5.52}$$

and the coarsest-scale wavelet space W_0 is defined as

$$W_0 := span\left\{\psi^1,\psi^2,\psi^3\right\} \tag{5.53}$$

In Ref. [4], it is demonstrated that once the wavelets that span W_0 are defined, the wavelets that span the higher-resolution spaces $\{W_j\}$, $j \geq 1$, can be generated through a recursive application of the formula

$$W_{j+1} := \bigoplus_{i=0}^{3} T_i W_j \tag{5.54}$$

where the spaces are nested:

$$W_{j+1} \supseteq W_j, \qquad j = 0,1,2,\ldots \tag{5.55}$$

Therefore, the wavelet spaces $\{W_j\}$ are constructed using the same operators as the scaling function spaces $\{V_j\}$. It should be noted that, since

$$V_{j+1} = V_j \oplus W_j$$

and $dim(V_j) = 4^j$, the dimension of each space W_j is given by

$$dim(W_j) = dim(V_{j+1}) - dim(V_j) = 4^{j+1} - 4^j = 3 \cdot 4^j \tag{5.56}$$

Because the wavelets that span the spaces $\{W_j\}$ are generated by applying the same technique that was used for the scaling functions, we define

$$\psi^r_{j,\mathbb{K}}(\xi, \eta) := 2^j (T_{k_j} \circ T_{k_{j-1}} \circ \cdots \circ T_{k_1} \circ \psi^r(\xi, \eta)), \quad r = 1, 2, 3 \tag{5.57}$$

Equation (5.57) is of the same form as equation (5.39), which defines an arbitrary scaling function $\phi_{j,\mathbb{K}}$. The 2^j term is a normalization factor and the multi-index $\mathbb{K}$ defines the sequence of mappings that generates each wavelet from the original wavelet ψ^r in W_0. The wavelet spaces $\{W_j\}$ are then defined as

$$W_j := span\left\{\psi^r_{j,\mathbb{K}} : \quad r = 1, 2, 3\right\}_{\mathbb{K} \in \Lambda_j} \tag{5.58}$$

We now have a multiresolution analysis over the domain Ω. Note that the wavelet complement spaces $\{W_j\}$ are orthogonal to one another. This can be demonstrated through a recursive application of the relationship

$$V_{j+1} = V_j \oplus W_j \tag{5.59}$$

where W_j is orthogonal to V_j. But equation (5.59) can also be written as

$$V_{j+1} = (V_{j-1} \oplus W_{j-1}) \oplus W_j \tag{5.60}$$

Since $W_{j-1} \subset V_j$ and V_j is orthogonal to W_j, it follows that W_{j-1} is orthogonal to W_j. Therefore, using the result from equation (5.35), we have the following orthogonal decomposition of $L^2(\Omega)$:

$$L^2(\Omega) = V_0 \oplus \left\{ \bigoplus_{j=0}^{\infty} W_j \right\} \tag{5.61}$$

5.2 Triangular Wavelet Decomposition and Reconstruction

To this point, a multiresolution analysis over the set Ω has been constructed using the method detailed by Micchelli and Xu [4]. Some of the tools that will enable us to apply this multiresolution analysis to the approximation of functions in $L^2(\Omega)$ can now be developed. First, the two-scale relationships, which determine how coarser-scale scaling functions and wavelets are obtained as weighted combinations of scaling functions on the next-finer scale, must be derived. Recall that the scaling function and wavelets on level 0 have already been expressed in terms of the scaling functions on level 1. We have the expressions

$$\phi(\xi,\eta) = \frac{1}{2}\phi_{1,0}(\xi,\eta) + \frac{1}{2}\phi_{1,1}(\xi,\eta) + \frac{1}{2}\phi_{1,2}(\xi,\eta) + \frac{1}{2}\phi_{1,3}(\xi,\eta) \tag{5.62}$$

$$\psi^1(\xi,\eta) = \frac{1}{\sqrt{2}}\phi_{1,0}(\xi,\eta) - \frac{1}{\sqrt{2}}\phi_{1,1}(\xi,\eta) \tag{5.63}$$

$$\psi^2(\xi,\eta) = \frac{1}{\sqrt{2}}\phi_{1,2}(\xi,\eta) - \frac{1}{\sqrt{2}}\phi_{1,3}(\xi,\eta) \tag{5.64}$$

$$\psi^3(\xi,\eta) = \frac{1}{2}\phi_{1,0}(\xi,\eta) + \frac{1}{2}\phi_{1,1}(\xi,\eta) - \frac{1}{2}\phi_{1,2}(\xi,\eta) - \frac{1}{2}\phi_{1,3}(\xi,\eta) \tag{5.65}$$

These relationships can be written in the form

$$\phi(\xi,\eta) = \sum_{p=0}^{3} a_p \phi_{1,p}(\xi,\eta) \tag{5.66}$$

$$\psi^r(\xi,\eta) = \sum_{p=0}^{3} b_p^r \phi_{1,p}(\xi,\eta), \quad r = 1,2,3 \tag{5.67}$$

where the scaling function filters $\{a_p\}$ and the wavelet filters $\{b_p^r\}$ are defined as

$$\{a_p\} = \{a_0, a_1, a_2, a_3\} = \left\{\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right\} \tag{5.68}$$

$$\{b_p^1\} = \{b_0^1, b_1^1, b_2^1, b_3^1\} = \left\{\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}, 0, 0\right\} \tag{5.69}$$

$$\{b_p^2\} = \{b_0^2, b_1^2, b_2^2, b_3^2\} = \left\{0, 0, \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right\} \tag{5.70}$$

$$\{b_p^3\} = \{b_0^3, b_1^3, b_2^3, b_3^3\} = \left\{\frac{1}{2}, \frac{1}{2}, -\frac{1}{2}, -\frac{1}{2}\right\} \tag{5.71}$$

Equations (5.66) and (5.67) represent the two-scale relationships between the scaling functions and wavelets on level 0 and the scaling functions on level 1. These equations can be generalized so that they apply over any two adjacent resolution levels. To show this, an arbitrary sequence of linear operators, $T_{k_1}, T_{k_2}, ..., T_{k_j}$, $k_i \in \{0, 1, 2, 3\}$, is applied to both sides of equations (5.66) and (5.67):

$$T_{k_j} \circ \cdots \circ T_{k_1} \circ \phi(\xi, \eta) = \sum_{p=0}^{3} a_p \left\{T_{k_j} \circ \cdots \circ T_{k_1} \circ \phi_{1,p}(\xi, \eta)\right\} \tag{5.72}$$

$$T_{k_j} \circ \cdots \circ T_{k_1} \circ \psi^r(\xi, \eta) = \sum_{p=0}^{3} b_p^r \left\{T_{k_j} \circ \cdots \circ T_{k_1} \circ \phi_{1,p}(\xi, \eta), \right\} \tag{5.73}$$

where the operators $\{T_i\}$ are as defined in equation (5.23). Note that equations (5.72) and (5.73) take advantage of the fact that each T_i is a linear operator. Recall that the scaling functions and wavelets on any given level j are defined as

$$\phi_{j,\mathbb{K}}(\xi, \eta) = 2^j (T_{k_j} \circ T_{k_{j-1}} \circ \cdots \circ T_{k_1} \circ \phi(\xi, \eta)) \tag{5.74}$$

$$\psi_{j,\mathbb{K}}^r(\xi, \eta) = 2^j (T_{k_j} \circ T_{k_{j-1}} \circ ... \circ T_{k_1} \circ \psi^r(\xi, \eta)) \tag{5.75}$$

Then, equations (5.72) and (5.73) can be written as

$$\phi_{j,\mathbb{K}}(\xi,\eta) = 2^j \sum_{p=0}^{3} a_p \left\{ T_{k_j} \circ \cdots \circ T_{k_1} \circ \phi_{1,p}(\xi,\eta) \right\}$$

$$\psi^r_{j,\mathbb{K}}(\xi,\eta) = 2^j \sum_{p=0}^{3} b^r_p \left\{ T_{k_j} \circ \cdots \circ T_{k_1} \circ \phi_{1,p}(\xi,\eta), \right\}$$

Note that

$$\phi_{1,p}(\xi,\eta) = 2T_p \circ \phi(\xi,\eta)$$

which implies that

$$\begin{aligned} T_{k_j} \circ \cdots \circ T_{k_1} \circ \phi_{1,p}(\xi,\eta) &= 2T_{k_j} \circ \cdots \circ T_{k_1} \circ T_p \circ \phi(\xi,\eta) \\ &= 2 \cdot 2^{-(j+1)} \phi_{j+1,p|\mathbb{K}}(\xi,\eta) \\ &= 2^{-j} \phi_{j+1,p|\mathbb{K}}(\xi,\eta) \end{aligned}$$

where $p|\mathbb{K}$ denotes the multi-index $pk_1k_2...k_j$, of length $j+1$. It follows that the generalized two-scale equations are given by

$$\phi_{j,\mathbb{K}}(\xi,\eta) = \sum_{p=0}^{3} a_p \phi_{j+1,p|\mathbb{K}}(\xi,\eta) \tag{5.76}$$

$$\psi^r_{j,\mathbb{K}}(\xi,\eta) = \sum_{p=0}^{3} b^r_p \phi_{j+1,p|\mathbb{K}}(\xi,\eta), \quad r = 1,2,3 \tag{5.77}$$

Any function $f \in L^2(\Omega)$ can be approximated as a linear combination of the scaling functions that span a fine-scale space V_j. The resulting approximation, denoted as f_j, is given by

$$f_j(\xi,\eta) = \sum_{\mathbb{K} \in \Lambda_j} \alpha_{j,\mathbb{K}} \phi_{j,\mathbb{K}}(\xi,\eta) \tag{5.78}$$

where $\{\alpha_{j,\mathbb{K}}\}$ represent constant scaling function coefficients. Equation (5.78) is a single-scale expansion of the function f on level j. Recalling that

$$V_j = V_{j-1} \oplus W_{j-1} \tag{5.79}$$

a two-scale expansion of f_j can be written as a linear combination of the scaling functions and wavelets on the coarser-resolution level $j-1$:

$$f_j(\xi,\eta) = \sum_{\mathbb{K}\in\Lambda_{j-1}} \alpha_{j-1,\mathbb{K}}\phi_{j-1,\mathbb{K}}(\xi,\eta) + \sum_{r=1}^{3}\sum_{\mathbb{K}\in\Lambda_{j-1}} \beta^r_{j-1,\mathbb{K}}\psi^r_{j-1,\mathbb{K}}(\xi,\eta) \tag{5.80}$$

Applying equation (5.79) recursively, we have the following decomposition of the subspace V_j:

$$V_j = V_0 \oplus W_0 \oplus W_1 \oplus \cdots \oplus W_{j-2} \oplus W_{j-1} \tag{5.81}$$

which leads to a multilevel representation of f_j of the form

$$f_j(\xi,\eta) = \alpha\phi(\xi,\eta) + \sum_{r=0}^{3}\sum_{l=0}^{j-1}\sum_{\mathbb{K}\in\Lambda_l} \beta^r_{l,\mathbb{K}}\psi^r_{l,\mathbb{K}} \tag{5.82}$$

Note that the first term on the right-hand side of equation (5.82) is the contribution of the scaling function ϕ that spans V_0.

Recall that the scaling functions on the same level are orthonormal, so that they satisfy

$$\int_\Omega \phi_{j,\mathbb{K}}(\xi,\eta)\,\phi_{j,\mathbb{P}}(\xi,\eta)\,d\Omega = \delta_{\mathbb{K},\mathbb{P}} \tag{5.83}$$

In addition, the scaling functions on any given level are orthogonal to the wavelets on levels of the same or finer resolution:

$$\int_{\Omega} \phi_{j,\mathbb{K}}(\xi,\eta)\,\psi^{r}_{l,\mathbb{P}}(\xi,\eta)\,d\Omega = 0 \qquad \forall\ \mathbb{K},\mathbb{P},\quad r\in\{1,2,3\},\quad j\le l \tag{5.84}$$

The wavelets on any given level are orthonormal to the wavelets on the same or any other level:

$$\int_{\Omega} \psi^{r}_{j,\mathbb{K}}(\xi,\eta)\,\psi^{s}_{l,\mathbb{P}}(\xi,\eta)\,d\Omega = \delta_{(j,r,\mathbb{K}),(l,s,\mathbb{P})} \tag{5.85}$$

The orthonormality properties listed in equations (5.83) through (5.85) greatly simplify many of the calculations that are made with the scaling functions and wavelets. To start, the scaling function coefficients in equation (5.78) can be computed by multiplying both sides of the equation by $\phi_{j,\mathbb{P}}$ and integrating over Ω. Making use of equation (5.83), we have

$$\alpha_{j,\mathbb{K}} = \int_{\Omega} f(\xi,\eta)\phi_{j,\mathbb{K}}(\xi,\eta)\,d\Omega \tag{5.86}$$

It is not difficult to derive relationships between the fine-scale coefficients that appear in the single-scale expansion of f in equation (5.78) and the multiscale coefficients that appear in equation (5.82). Towards this goal, equations (5.78) and (5.80) can be combined into the expression

$$\sum_{\mathbb{K}} \alpha_{j,\mathbb{K}}\phi_{j,\mathbb{K}}(\xi,\eta) = \sum_{\mathbb{K}} \alpha_{j-1,\mathbb{K}}\phi_{j-1,\mathbb{K}}(\xi,\eta) + \sum_{r=1}^{3}\sum_{\mathbb{K}} \beta^{r}_{j-1,\mathbb{K}}\psi^{r}_{j-1,\mathbb{K}}(\xi,\eta) \tag{5.87}$$

Suppose that, knowing the finer-scale expansion coefficients $\{\alpha_{j,\mathbb{K}}\}$, we wish to calculate the coarser-scale coefficients $\{\alpha_{j-1,\mathbb{K}}\}$ and $\{\beta^{r}_{j-1,\mathbb{K}}\}$. A decomposition formula for the scaling function coefficients $\{\alpha_{j-1,\mathbb{K}}\}$ can be obtained by multiplying both sides

of equation (5.87) by the scaling function $\phi_{j-1,\mathbb{P}}$ and integrating over Ω. The orthonormality properties in equations (5.83) and (5.84) enable the resulting equation to be simplified to the form

$$\alpha_{j-1,\mathbb{P}} = \sum_{\mathbb{K}} \alpha_{j,\mathbb{K}} \int_{\Omega} \phi_{j,\mathbb{K}}(\xi,\eta)\phi_{j-1,\mathbb{P}}(\xi,\eta)d\Omega \tag{5.88}$$

The two-scale relationship given in equation (5.76) can be used to substitute for $\phi_{j-1,\mathbb{P}}$ in equation (5.88). Then, we obtain

$$\alpha_{j-1,\mathbb{P}} = \sum_{\mathbb{K}} \sum_{m=0}^{3} a_m \alpha_{j,\mathbb{K}} \int_{\Omega} \phi_{j,\mathbb{K}}(\xi,\eta)\phi_{j,m|\mathbb{P}}(\xi,\eta)d\Omega \tag{5.89}$$

Once again using fact that the scaling functions are orthonormal, equation (5.89) simplifies to

$$\alpha_{j-1,\mathbb{P}} = \sum_{\mathbb{K}} \sum_{m=0}^{3} a_m \alpha_{j,\mathbb{K}} \delta_{\mathbb{K},m|\mathbb{P}}$$

Finally, the following decomposition formula is obtained:

$$\alpha_{j-1,\mathbb{P}} = \sum_{m=0}^{3} a_m \alpha_{j,m|\mathbb{P}} \tag{5.90}$$

In a similar manner, decomposition formulas for the coarser-scale wavelet coefficients can be derived by multiplying both sides of equation (5.87) by the wavelet $\psi^r_{j-1,\mathbb{P}}$ and integrating over Ω. The orthonormality properties and the two-scale relationships given in equation (5.77) lead to the result

$$\beta^r_{j-1,\mathbb{P}} = \sum_{m=0}^{3} b^r_m \alpha_{j,m|\mathbb{P}}, \qquad r = 1,2,3 \tag{5.91}$$

Note that equations (5.90) and (5.91) can be applied recursively. For example, if the scaling function coefficients $\{\alpha_{j,\mathbb{K}}\}$ on some level j are known, they can be used to calculate the scaling function and wavelet coefficients on level $j-1$. Then, applying

equations (5.90) and (5.91) again, the scaling function coefficients on level $j-1$ can be used to calculate the scaling function and wavelet coefficients on level $j-2$. This process can be continued over however many levels there are in the multilevel expansion of the function f in equation (5.82). In this manner, the entire discrete wavelet transform can be computed efficiently.

Next, suppose that the reverse operation is needed. That is, given scaling function and wavelet coefficients on some level $j-1$, we seek to obtain the finer-resolution scaling function coefficients on level j, $\{\alpha_{j,\mathbb{K}}\}$. A reconstruction formula can be derived that will enable the calculation of these finer-scale coefficients. This formula is obtained by multiplying both sides of equation (5.87) by the scaling function $\phi_{j,\mathbb{P}}$ and integrating over Ω. We obtain

$$\begin{aligned}\alpha_{j,\mathbb{P}} &= \sum_{\mathbb{K}} \alpha_{j-1,\mathbb{K}} \int_{\Omega} \phi_{j-1,\mathbb{K}}(\xi,\eta)\phi_{j,\mathbb{P}}(\xi,\eta)d\Omega \\ &\quad + \sum_{r=1}^{3} \sum_{\mathbb{K}} \beta^{r}_{j-1,\mathbb{K}} \int_{\Omega} \psi^{r}_{j-1,\mathbb{K}}(\xi,\eta)\phi_{j,\mathbb{P}}(\xi,\eta)d\Omega\end{aligned} \tag{5.92}$$

Substituting for $\phi_{j-1,\mathbb{K}}$, $\psi^1_{j-1,\mathbb{K}}$, $\psi^2_{j-1,\mathbb{K}}$, and $\psi^3_{j-1,\mathbb{K}}$ using the two-scale equations, equation (5.92) takes the form

$$\begin{aligned}\alpha_{j,\mathbb{P}} &= \sum_{\mathbb{K}} \sum_{m=0}^{3} a_m \alpha_{j-1,\mathbb{K}} \int_{\Omega} \phi_{j,m|\mathbb{K}}(\xi,\eta)\phi_{j,\mathbb{P}}(\xi,\eta)d\Omega \\ &\quad + \sum_{r=1}^{3} \sum_{\mathbb{K}} \sum_{m=0}^{3} b^{r}_{m} \beta^{r}_{j-1,\mathbb{K}} \int_{\Omega} \phi_{j,m|\mathbb{K}}(\xi,\eta)\phi_{j,\mathbb{P}}(\xi,\eta)d\Omega\end{aligned}$$

Then, taking advantage of the orthonormality properties of the scaling functions, this reduces to

$$\alpha_{j,\mathbb{P}} = \sum_{\mathbb{K}} \sum_{m=0}^{3} \left\{ a_m \alpha_{j-1,\mathbb{K}} + b^1_m \beta^1_{j-1,\mathbb{K}} + b^2_m \beta^2_{j-1,\mathbb{K}} + b^3_m \beta^3_{j-1,\mathbb{K}} \right\} \delta_{m|\mathbb{K},\mathbb{P}}$$

Finally, the following reconstruction formula is obtained:

$$\alpha_{j,m[\mathbb{P}} = a_m \alpha_{j-1,\mathbb{P}} + \sum_{r=1}^{3} b_m^r \beta_{j-1,\mathbb{P}}^r \tag{5.93}$$

Equation (5.93) can be used to calculate the scaling function coefficients on some level j from the coarser-resolution scaling function and wavelet coefficients on level $j-1$. Of course, it can be applied recursively to perform reconstruction over multiple levels in calculating the inverse discrete wavelet transform.

CHAPTER 6
WAVELET-BASED IDENTIFICATION OF VOLTERRA KERNELS

The first five chapters have presented the necessary background for applying wavelets to the identification of Volterra kernels. In this chapter, two wavelet-based kernel identification algorithms are described in detail. The first implementation uses the triangular wavelets derived in Chapter 5 to identify the triangular form of the second-order kernel. This algorithm also employs the Haar wavelet for the representation of the first-order kernel. The second wavelet implementation uses the piecewise-polynomial multiwavelets constructed in Chapter 4 to identify first and second-order kernels. In this case, the symmetric form of the second-order kernel is identified. Before these wavelet-based algorithms are discussed, a survey of kernel identification techniques is given in Section 6.1. Many approaches to kernel identification, including the wavelet implementations discussed here, lead to a matrix formulation in which a least-squares solution must be obtained for a vector of kernel coefficients. Because kernel identification is an inverse problem that is frequently ill-posed, regularization techniques are often needed to obtain stable solutions to the least-squares problem. This is discussed in Section 6.4 as it is a critical issue in the implementation of the wavelet algorithms. Finally, in Section 6.5, some of the finer details and issues concerning the wavelet implementations are reviewed for the interested reader.

6.1 Kernel Identification Techniques

In the analysis of nonlinear dynamical systems, the goal is to identify the kernels that appear in the Volterra series representation of the system output. As discussed earlier, these kernels characterize a given system since they comprise the operators

that map the input into the output. There are many approaches to the kernel identification problem. As discussed earlier, when the system is described in terms of an ODE, the kernels can be derived analytically. Another approach is to discretize the Volterra series and solve a least-squares problem for the discrete kernel coefficients. In some cases, specific inputs are given to the system in order to measure the Volterra kernels in the time or frequency domain. Other methods are statistical in nature and involve the use of Gaussian white noise inputs. In many approaches, including the wavelet algorithms discussed later in this chapter, the kernels are expanded in terms of a set of basis functions. The coefficients in the representation of the kernels must then be determined. These standard approaches and others are reviewed in this section.

A natural approach to kernel identification is to simply discretize the continuous-time Volterra series and solve for the discrete kernel coefficients. The discrete-time representation of the Volterra series for a stationary, causal, nonlinear dynamical system is of the form [15]

$$y(k) = \sum_{n=1}^{\infty} \sum_{i_1=0}^{k} \cdots \sum_{i_n=0}^{k} h_n(i_1, \ldots, i_n) u(k - i_1) \cdots u(k - i_n) \tag{6.1}$$

In equation (6.1), it is assumed that the input and output sequences range from $k = 0$ to N. Then, equation (6.1) yields $N + 1$ equations for the discrete outputs. Furthermore, it is assumed that the initial conditions are zero so that there is no zero-order term. Of course, in practice one cannot identify an infinite number of Volterra kernels, and the series of operators is truncated at some integer M. As an example, assume that the series has been truncated to include only the first and second-order operators ($M = 2$). An output vector $\underline{y}$ and a vector of discrete kernel coefficients $\underline{h}$ can be defined as

$$\underline{y} = [y(0) \cdots y(N)]^T$$

$$\underline{h} = [h_1(0)\ h_1(1)\cdots h_1(N)\ h_2(0,0)\ h_2(0,1)\cdots h_2(0,N)$$
$$h_2(1,0)\cdots h_2(1,N)\cdots h_2(N,0)\cdots h_2(N,N)]^T$$

Then, defining a matrix of input values

$$[U]^T = \begin{bmatrix}
u(0) & u(1) & \cdots & u(N) \\
0 & u(0) & & u(N-1) \\
0 & 0 & \cdots & u(N-2) \\
\vdots & \vdots & & \vdots \\
0 & 0 & \cdots & u(0) \\
u^2(0) & u^2(1) & & u^2(N) \\
0 & u(1)\,u(0) & \cdots & u(N)\,u(N-1) \\
0 & 0 & & u(N)u(N-2) \\
\vdots & \vdots & \cdots & \vdots \\
0 & 0 & & u(N)u(0) \\
0 & u(0)\,u(1) & \cdots & u(N-1)\,u(N) \\
0 & u^2(0) & & u^2(N-1) \\
0 & 0 & \cdots & u(N-1)\,u(N-2) \\
\vdots & \vdots & & \vdots \\
0 & 0 & \cdots & u(N-1)u(0) \\
\vdots & \vdots & & \vdots \\
0 & 0 & \cdots & u(0)u(N) \\
\vdots & \vdots & & \vdots \\
0 & 0 & \cdots & u^2(0)
\end{bmatrix}$$

the $N+1$ discrete output equations given by equation (6.1) can be written in matrix form as

$$\underline{y} = [U]\ \underline{h} \tag{6.2}$$

In general, $[U]$ is a $(N+1)$ x $\left(\sum_{n=1}^{M}(N+1)^n\right)$ matrix, where $N+1$ is the number of discrete inputs and M represents the highest-order operator included in the Volterra series representation. Equation (6.2) can be solved for the values of the discrete kernel coefficients in $\underline{h}$ using a least-squares method. Unfortunately, with this approach, the higher-order nonlinear kernels are represented in terms of a prohibitively large number of discrete coefficients. In general, if the input and output sequences are of length $N+1$, the discrete nth-order Volterra kernel is composed of $N_n = (N+1)^n$ terms that must be identified. This is a serious limitation of this simple discrete approach. The number of coefficients in the discrete model can be reduced significantly by assuming truncated forms of the Volterra kernels. Because most physical systems have finite memory, meaning that a current input will have an influence on future outputs for a finite time, the corresponding Volterra kernels can be expected to decay to zero in a finite time. This is consistent with the concept of fading memory introduced in Section 2.5. For example, for a linear system, the memory length represents the time it takes for the impulse response function (or equivalently, the first-order kernel) to decay to zero. With the assumption of finite memory, the first-order kernel can be expressed in terms of $\hat{N}_1 = f_s T_1 + 1$ unknown coefficients, where f_s is the sampling frequency of the input/output data and T_1 is the time at which the kernel has been truncated. Similarly, truncating the second-order kernel at time T_2, the number of discrete second-order kernel coefficients can be reduced to $\hat{N}_2 = (f_s T_2 + 1)^2$. This can result in a significant reduction in the number of coefficients that must determined in equation (6.2), especially if the input/output data sets are very large. Even so, the number of coefficients is still typically very large. Therefore, the simple discrete approach described above has limited utility in practice.

Another approach to kernel identification is to use impulsive inputs to measure the kernels of time-invariant systems [15], [16]. This is a generalization of the fact that the first-order kernel for a linear system is equivalent to the impulse response of the system. This can easily be demonstrated by evaluating the output of the first-order Volterra operator for a unit impulse input δ_0, defined as a function of infinitesimal width and unit area centered at the origin. The first-order response to this input is given by

$$\begin{aligned} y(t) &= \int_0^t h_1(t-\xi)u(\xi)d\xi \\ &= \int_0^t h_1(t-\xi)\delta_0(\xi)d\xi \\ &= h_1(t) \end{aligned} \tag{6.3}$$

where the sifting property of the impulse function has been employed. Now consider a weakly nonlinear system that can be described in terms of the first and second-order Volterra kernels. The response y_0 of the system to the impulse input δ_0 is given by

$$\begin{aligned} y_0(t) &= \int_0^t h_1(t-\xi)\delta_0(\xi)d\xi \\ &\quad + \int_0^t \int_0^t h_2(t-\xi, t-\eta)\delta_0(\xi)\delta_0(\eta)d\xi d\eta \\ &= h_1(t) + h_2(t,t) \end{aligned} \tag{6.4}$$

It should be noted, then, that the response y_0 of the nonlinear system to an impulse input is not equivalent to the first-order kernel but rather it is given as the sum of the first-order kernel and the diagonal component of the second-order kernel. The components of the second-order kernel, as well as the first-order kernel, can be measured by applying double impulses of the form

$$u_1(t) = \delta_0(t) + \delta_0(t - T) \tag{6.5}$$

It is not difficult to show that the response y_1 to this input is given by

$$y_1(t) = h_1(t) + h_1(t - T) + h_2(t, t) + 2h_2(t, t - T) + h_2(t - T, t - T) \tag{6.6}$$

where $h_2(t - T, t) = h_2(t, t - T)$ due to the symmetry of the second-order kernel. Note that, due to the time-invariance of the system, the response to the time-shifted impulse $\delta_0(t - T)$ is simply $y_0(t - T)$, given as

$$y_0(t - T) = h_1(t - T) + h_2(t - T, t - T) \tag{6.7}$$

Then, using equations (6.4) and (6.7), equation (6.6) can be written as

$$y_1(t) = y_0(t) + y_0(t - T) + 2h_2(t, t - T) \tag{6.8}$$

Then, the component $h_2(t, t - T)$ of the second-order kernel is given by

$$h_2(t, t - T) = \tfrac{1}{2}\left(y_1(t) - y_0(t) - y_0(t - T)\right) \tag{6.9}$$

In the square domain of support of the second-order kernel, the component $h_2(t, t-T)$ can be interpreted as the value of the kernel along the line $\eta = \xi - T$. Equation (6.9) implies that, in theory, one can measure a given component of the second-order kernel by applying the inputs δ_0 and u_1 to the system. The measured responses y_0 and y_1 can then be used to calculate the component $h_2(t, t-T)$. By varying the time delay T, other components of the second-order kernel can be measured. As noted by Silva [27], the first-order kernel can be extracted by applying the input $2\delta_0$. The response y_2 to this input is

$$y_2(t) = 2h_1(t) + 4h_2(t,t) \tag{6.10}$$

Then, combining this result with equation (6.4), the first-order kernel can be obtained as

$$h_1(t) = 2y_0(t) - \tfrac{1}{2}y_2(t) \tag{6.11}$$

It is straightforward to extend this approach to include the contributions of higher-order kernels. The difficulty with this technique is that, for many systems, it is impossible to apply an accurate impulse input. For example, there is no way to deliver an accurate impulse to a test section in a wind tunnel or an aircraft during a flight test. This method has been applied in some cases, however. For instance, Tawfiq and Vinh [102] have applied this technique to study structural nonlinearities. In order to do this, they developed a special hammer that is capable of delivering multiple impulses over short time intervals. Silva [27,28] has successfully measured first and second-order Volterra kernels for nonlinear aeroelastic systems by applying discrete-time unit impulses to CFD models of these systems. In related work, Raveh [29,30] has investigated the use of step inputs in CFD models to identify step-type kernels for nonlinear aerodynamic systems.

One of the most common approaches to kernel identification is the harmonic probing technique, in which the kernels are measured in the frequency domain through the use of harmonic inputs. Consider, once again, a weakly nonlinear system that is modeled in terms of the first and second-order Volterra operators. The response of this system to a harmonic input $e^{j\Omega t}$ is given by

$$y(t) = \int_0^t h_1(\xi) e^{j\Omega(t-\xi)} d\xi$$

$$
\begin{aligned}
&+\int_0^t \int_0^t h_2(\xi,\eta) e^{j\Omega(t-\xi)} e^{j\Omega(t-\eta)} d\xi d\eta \\
= &\int_0^t h_1(\xi) e^{-j\Omega\xi} d\xi \; e^{j\Omega t} \\
&+\int_0^t \int_0^t h_2(\xi,\eta) e^{-j\Omega\xi} e^{-j\Omega\eta} d\xi d\eta \; e^{2j\Omega t}
\end{aligned} \tag{6.12}
$$

Recall from Chapter 2 that the frequency response functions of the first and second-order kernels, also known as Volterra kernel transforms, are defined as

$$
H_1(j\omega) = \int_{-\infty}^{\infty} h_1(\xi) e^{-j\omega\xi} d\xi \tag{6.13}
$$

$$
H_2(j\omega_1, j\omega_2) = \int_{-\infty}^{\infty} h_2(\xi,\eta) e^{-j(\omega_1\xi+\omega_2\eta)} d\xi d\eta \tag{6.14}
$$

Then, equation (6.12) can be written as

$$
y(t) = H_1(j\Omega) e^{j\Omega t} + H_2(j\Omega, j\Omega) e^{2j\Omega t} \tag{6.15}
$$

In this manner, the first-order kernel transform H_1 can be determined over any range of frequencies by varying the input frequency. At the same time, only diagonal components of H_2 in the (ω_1, ω_2) frequency domain can be measured. In order to determine H_2 at off-diagonal points, a dual-tone harmonic signal of the form

$$
u(t) = e^{j\Omega_1 t} + e^{j\Omega_2 t} \tag{6.16}
$$

is employed. The output corresponding to this input is given by

$$
\begin{aligned}
y(t) \;=\; & H_1(j\Omega_1) e^{j\Omega_1 t} + H_1(j\Omega_2) e^{j\Omega_2 t} + H_2(j\Omega_1, j\Omega_1) e^{j2\Omega_1 t} \\
& + 2H_2(j\Omega_1, j\Omega_2) e^{j(\Omega_1+\Omega_2)t} + H_2(j\Omega_2, j\Omega_2) e^{j2\Omega_2 t}
\end{aligned} \tag{6.17}
$$

Then, it is possible to determine the value of the second-order frequency response H_2 at the point $(j\Omega_1, j\Omega_2)$ as half the magnitude of the response at the frequency $\Omega_1 + \Omega_2$. It is not difficult to generalize these results to include the measurement of higher-order kernels. The harmonic probing, or multitone method, was first introduced by Bedrosian and Rice [103]. Worden *et al.* [104] extended the technique to measure the cross-kernels appearing in the multi-input Volterra series. A drawback of harmonic probing is that measuring the kernels using dual tone signals can be very time consuming as many tests are required to extract the kernels. To address this concern, Boyd and Chua [59] employed input signals composed of several harmonics for the purposes of measuring the kernels at multiple points for each input. The input signals were designed such that as few output frequencies as possible would correspond to the input frequencies. An output frequency that matches the input frequency is useless for measuring higher-order kernels since there is always a contribution from the first-order kernel at that frequency. Evans *et al.* [105] and Weiss *et al.* [106] further improved on this method, designing input signals that maximize the number of distinct points in the kernel that can be measured with one input. In addition, they sought to achieve a nearly even spacing between measured points while minimizing the largest frequency contribution. One limitation of these multitone methods is that they require specific inputs that may not be feasible in a noisy experimental environment.

Another classical kernel identification technique is the cross-correlation method developed by Lee and Schetzen [58]. In this technique, the kernels associated with the Wiener series are estimated instead of Volterra kernels. The Wiener series corresponds to an orthogonalization of the Volterra series via the Gram-Schmidt procedure under the assumption of a Gaussian white noise input [13]. The advantage of the Wiener series is that the terms in the series are orthogonal and can be estimated one at a time. In contrast, the Volterra series is not orthogonal and the Volterra kernels must

be identified simultaneously. However, a major drawback of the Wiener series is that, unlike the Volterra series, the kernels are input-dependent. The first few terms in the Wiener series are given as [58]

$$G_0[k_0, u(t)] = k_0 \tag{6.18}$$

$$G_1[k_1, u(t)] = \int_0^t k_1(\xi)u(t-\xi) \tag{6.19}$$

$$G_2[k_2, u(t)] = \int_0^t \int_0^t k_2(\xi,\eta)u(t-\xi)u(t-\eta)d\xi d\eta - \sigma_u^2 \int_0^t k_2(\eta,\eta)d\eta \tag{6.20}$$

where σ_u^2 denotes the variance of the input. The Wiener operators of second order and higher are directly dependent on the input variance. The cross-correlation method is usually applied as a discrete-time procedure. First, as noted in Ref. [96], the zero-order Wiener kernel is a constant equal to the output mean μ_y:

$$k_0 = \frac{1}{N}\sum_{t=1}^{N} y(t) = \mu_y \tag{6.21}$$

where N is the length of the data record. The first-order Wiener kernel can be shown to be proportional to the first-order cross-correlation ϕ_{uy} between the input and output. In practice, however, the first-order kernel is defined in terms of the first-order cross-correlation between the input and the zero-order residual, defined as [96]

$$\nu_0(t) = y(t) - k_0 \tag{6.22}$$

Then, the first-order kernel can be obtained as

$$k_1(\tau) = \frac{1}{\sigma_u^2}\phi_{u\nu_0}(\tau) = \frac{1}{N\sigma_u^2}\sum_{t=\tau+1}^{N} u(t-\tau)\nu_0(t) \tag{6.23}$$

Similarly, the second-order Wiener kernel is proportional to the second-order cross-correlation $\phi_{uu\nu_1}$ between the input and the first-order residual [96]:

$$k_2(\tau_1, \tau_2) = \frac{1}{2\sigma_u^4}\phi_{uu\nu_1}(\tau_1, \tau_2) = \frac{1}{2N\sigma_u^4} \sum_{t=1+\max(\tau_1,\tau_2)}^{N} u(t-\tau_1)u(t-\tau_2)\nu_1(t) \quad (6.24)$$

where the first-order residual is given by

$$\nu_1(t) = \nu_0(t) - \sum_{\tau=0}^{N-1} k_1(\tau)u(t-\tau) \quad (6.25)$$

In general, the nth-order kernel is proportional to the nth-order cross-correlation between the input u and the $(n-1)$th-order residual ν_{n-1} [96]. As noted by several authors, a disadvantage of the cross-correlation method is that it typically requires a very large data set in order to obtain accurate kernel estimates. Furthermore, the technique requires a Gaussian white noise input, which is not always possible in practice.

The impulse input and harmonic probing techniques are examples of direct approaches to kernel identification in which the kernels are measured in terms of the system response to specific inputs. The cross-correlation method is a statistical method that also requires a specific type of input. In contrast, with indirect methods such as the simple discrete model described earlier, the kernels are identified from input/output data from the system. Many indirect approaches entail expanding the kernels in terms of a set of basis functions. Then, the kernel identification problem becomes one of solving for the expansion coefficients that represent the kernels. A popular choice of basis is the Laguerre polynomials, which were first suggested by Wiener [13] for the representation of Wiener kernels. They were later employed by Marmarelis [60] in the identification of kernels for biological systems. The discrete Laguerre functions take the form [60]

$$b_j(\tau) = \alpha^{(\tau-j)/2}(1-\alpha)^{1/2}\sum_{k=0}^{j}(-1)^k\binom{\tau}{k}\binom{j}{k}\alpha^{j-k}(1-\alpha)^k, \quad \tau \geq 0 \tag{6.26}$$

where j is the order of the Laguerre function. The parameter α, $0 < \alpha < 1$, determines the exponential rate of decay of these functions. As noted by Marmarelis [60], the choice of α is critical to the effectiveness of the Laguerre approach. As a basic rule, as α increases, the Laguerre functions become more dilated and are better suited for approximating kernels with longer memory. Similar to this approach, Reisenthel [34] used decaying exponentials to approximate the first and second-order kernels for nonlinear aeroelastic systems. Korenberg and Hunter [107] have noted that using smooth basis functions such as the Laguerre polynomials almost always results in smooth kernel estimates. Therefore, they may not be effective in approximating kernels with sharp features. Nikolaou and Mantha [61] applied biorthogonal wavelets to compress the first and second-order kernels that model a particular chemical reaction. They used a priori knowledge about the plant to determine which wavelet coefficients to retain in the model. Kurdila *et al.* [62] also used biorthogonal wavelets for the approximation of Volterra kernels for an experimental aeroelastic system designed to study limit cycle oscillation in aircraft. The wavelet-based kernel identification algorithms described later in this chapter clearly fall into the same category as these methods. The advantage of expanding the kernels in terms of a basis set is that there is no specific input excitation required. Therefore, these methods can typically use physically realizable inputs. A drawback of these methods is that they often lead to ill-posed problems, as discussed in Section 6.4.

There are numerous other approaches to kernel identification. Sometimes, the system is assumed to have a specific block structure. For example, the Hammerstein model assumes that the system is composed of a static nonlinearity followed by a linear dynamical system. Conversely, the Wiener model assumes that the structure of

the system corresponds to a linear dynamical system followed by a static nonlinearity. Korenberg [108] developed a parallel cascade algorithm whereby the system is modeled by a finite number of Wiener structures connected in parallel. Korenberg [109] also contributed a fast orthogonal identification method that is frequently used in biomedical engineering applications. Neural networks provide yet another tool for the estimation of Volterra kernels [110], [63]. Wray and Green [63] have demonstrated that many artificial neural networks have equivalent finite Volterra series representations. As a final example, adaptive Volterra filters are widely used in electrical engineering applications. The discrete filter coefficients are often computed recursively using the least mean squares (LMS) approach or the recursive least-squares (RLS) method. The reader can refer to the texts by Rugh [15] and Schetzen [16] for more discussion of kernel identification techniques. In addition, Korenberg and Hunter [107] have written a survey paper on kernel identification, and Westwick and Kearney [96] have conducted an extensive review of nonparametric system identification.

6.2 Kernel Identification Using Triangular Wavelets

In this section, the first of two wavelet-based kernel identification algorithms is described in detail. In both cases, it is assumed that we have a single-input/single-output system with weak nonlinearities. More specifically, it is assumed that the output can be expressed in terms of the first and second-order Volterra operators as

$$y(t) = y_1(t) + y_2(t) \tag{6.27}$$

Furthermore, the discussion is limited to time-invariant, causal systems with one-sided inputs (that is, the input starts at time $t = 0$). Recall that, under these conditions, the outputs of the first and second-order Volterra operators take the form

$$y_1(t) = \int_0^t h_1(\xi)u(t-\xi)d\xi \tag{6.28}$$

$$y_2(t) = \int_0^t \int_0^t h_2(\xi, \eta) u(t-\xi) u(t-\eta) d\xi d\eta \tag{6.29}$$

In this section, the triangular wavelets derived in Chapter 5 are used to approximate the triangular form of the second-order kernel. Using the triangular form of the kernel, the output of the second-order operator can be written as

$$y_2(t) = \int_0^t \int_0^\eta h_{2,tri}(\xi, \eta) u(t-\xi) u(t-\eta) d\xi d\eta \tag{6.30}$$

where $h_{2,tri}$ denotes the triangular second-order kernel. A wavelet basis must also be chosen for the representation of the first-order kernel. The Haar wavelet is a reasonable choice since, like the triangular wavelets, it is a piecewise-constant function. In the next section, the piecewise-polynomial multiwavelets derived in Chapter 4 are used to approximate the first-order kernel and the symmetric form of the second-order kernel. Both wavelet-based kernel identification algorithms result in a linear least-squares problem that must be solved for the wavelet coefficients that represent the kernels.

In order to derive a discrete form of the first-order output in equation (6.28), the input is first discretized by employing a zero-order hold, which is a piecewise-constant approximation of the input. In a zero-order hold approximation, each time the input is sampled, it is held constant until the next sample is taken. Recall from Chapter 3 that the Haar scaling function, also known as the characteristic function of the interval $[0, 1]$, is defined as

$$\chi(\xi) := \begin{cases} 1 & 0 \leq \xi \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{6.31}$$

Then, a zero-order hold representation of the input can be written in terms of the Haar scaling function as

$$u_j(\xi) = \sum_{k=0}^{N-1} u_{j,k}\chi_{j,k}(\xi) \tag{6.32}$$

where we have the usual notation

$$\chi_{j,k}(\xi) = 2^{j/2}\chi(2^j\xi - k) \tag{6.33}$$

and N denotes the number of samples taken. The dilation index j defines the sampling step of the input to be $\Delta t = 2^{-j}$. The coefficients $\{u_{j,k}\}$ in equation (6.32) are scaled samples of the input:

$$u_{j,k} = 2^{-j/2}u(2^{-j}k), \qquad k = 0, \ldots, N-1 \tag{6.34}$$

The output is also discretized with a sampling step of 2^{-j}. Given N samples of the output, equation (6.27) yields N equations for the N discrete first-order outputs:

$$y_{1,j}(t_n) = y_{1,j}(2^{-j}n) = \int_0^{t_n} h_1(\xi)u_j(t_n - \xi)d\xi, \qquad n = 1, \ldots, N \tag{6.35}$$

Note that the subscript j on the input and output denotes an approximation of these functions on resolution level j. From equation (6.32), the input $u_j(t_n - \xi)$ can be written as

$$u_j(t_n - \xi) = u_j(2^{-j}n - \xi) = \sum_{k=0}^{n-1} u_{j,k}\chi_{j,k}(t_n - \xi) \tag{6.36}$$

Note that, from the definition of the characteristic function χ and equation (6.33),

$$\chi_{j,k}(t_n - \xi) = \begin{cases} 1 & 2^{-j}k \le 2^{-j}n - \xi \le 2^{-j}(k+1) \\ 0 & \text{otherwise} \end{cases}$$

$$= \begin{cases} 1 & 2^{-j}(k-n) \le -\xi \le 2^{-j}(k-n+1) \\ 0 & \text{otherwise} \end{cases}$$

$$= \begin{cases} 1 & 2^{-j}(n-k-1) \le \xi \le 2^{-j}(n-k) \\ 0 & \text{otherwise} \end{cases}$$

$$= \chi_{j,n-k-1}(\xi) \tag{6.37}$$

Therefore, we have

$$u_j(t_n - \xi) = \sum_{k=0}^{n-1} u_{j,k}\chi_{j,n-k-1}(\xi) \tag{6.38}$$

The next step is to approximate the first-order kernel in terms of the Haar wavelet basis. A single-scale expansion of the kernel using Haar scaling functions takes the form

$$h_{1,j}(\xi) = \sum_{p=0}^{N-1} \alpha_{j,p}\chi_{j,p}(\xi) \tag{6.39}$$

where $h_{1,j}$ denotes an approximation of the kernel on level j and $\{\alpha_{j,p}\}$ are scaling function coefficients. This single-scale representation is simply a piecewise-constant approximation of the first-order kernel. Strictly speaking, the first-order kernel is supported over the entire domain $[0, t_N]$, where $t_N = 2^{-j}N$ represents the time duration of the data set. Therefore, the summation in equation (6.39) extends from $p = 0$ to $N - 1$, where N is the total number of input/output samples. In practice, however, many physical systems possess finite memory in that the present input influences future outputs for a finite amount of time. This implies that the Volterra kernels of the system decay to zero in a finite amount of time. Assuming that the first-order kernel decays to zero in T_1 seconds, the upper summation in equation (6.39) can be truncated to $N_1 - 1$, where $N_1 = 2^{-j}T_1$. The assumption of a finite kernel greatly reduces the number of unknown coefficients that must be identified, especially when the input/output data set is large. For example, consider a case where 16 seconds of

data have been sampled at resolution level $j = 8$, or a sampling rate of 256 Hz. Then, there would be a total of 4096 samples of the input and output and the representation of a 16-second first-order kernel would require 4096 coefficients in equation (6.39). If, however, the kernel has a memory of only 2 seconds, the number of unknown coefficients can be reduced to 512. Of course, one does not usually know the memory length of the kernel *a priori* and an assumption must be made regarding the kernel size. More will be said about this later, but for now it is assumed that the first-order kernel has a memory length of T_1 seconds.

Substituting the truncated kernel in equation (6.39) and the expression for the input in equation (6.38) into equation (6.35), we obtain

$$y_{1,j}(t_n) = \sum_{p=0}^{N_1-1} \sum_{k=0}^{n-1} u_{j,k}\alpha_{j,p} \int_0^{T_1} \chi_{j,n-k-1}(\xi)\chi_{j,p}(\xi)d\xi \tag{6.40}$$

The upper limit of integration has been lowered to T_1 since the kernel has been truncated. By the orthonormality of the Haar scaling functions, we have

$$\int_0^{T_1} \chi_{j,n-k-1}(\xi)\chi_{j,p}(\xi)d\xi = \delta_{n-k-1,p} = \delta_{n-p-1,k} \tag{6.41}$$

Then, equation (6.40) reduces to

$$y_{1,j}(t_n) = \sum_{p=0}^{N_1-1} u_{j,n-p-1}\alpha_{j,p} \tag{6.42}$$

To be more precise, the upper limit in this summation is $n-1$ for $n < N_1$ and $N_1 - 1$ for $n \geq N_1$. Equation (6.42) provides N equations to be solved for the N_1 coefficients $\{\alpha_{j,p}\}_{p=0}^{N_1-1}$ that represent the first-order kernel:

$$\begin{aligned} y_{1,j}(t_1) &= u_{j,0}\alpha_{j,0} \\ y_{1,j}(t_2) &= u_{j,1}\alpha_{j,0} + u_{j,0}\alpha_{j,1} \end{aligned} \tag{6.43}$$

$$\vdots$$

$$y_{1,j}(t_N) = u_{j,N-1}\alpha_{j,0} + u_{j,N-2}\alpha_{j,1} + \cdots + u_{j,N-N_1}\alpha_{j,N_1-1}$$

Defining the vectors

$$\underline{y}_{1,j} := \begin{Bmatrix} y_{1,j}(t_1) \\ \vdots \\ y_{1,j}(t_N) \end{Bmatrix} \qquad \underline{\alpha}_j := \begin{Bmatrix} \alpha_{j,0} \\ \vdots \\ \alpha_{j,N_1-1} \end{Bmatrix}$$

and the N x N_1 input matrix

$$[U_1] = \begin{bmatrix} u_{j,0} & 0 & 0 & \cdots & 0 \\ u_{j,1} & u_{j,0} & 0 & \cdots & 0 \\ \vdots & & & & \vdots \\ u_{j,N_1-1} & u_{j,N_1-2} & u_{j,N_1-3} & \cdots & u_{j,0} \\ u_{j,N_1} & u_{j,N_1-1} & u_{j,N_1-2} & \cdots & u_{j,1} \\ \vdots & & & & \vdots \\ u_{j,N-1} & u_{j,N-2} & u_{j,N-3} & \cdots & u_{j,N-N_1} \end{bmatrix}$$

the N expressions in equation (6.43) can be written in matrix form as

$$\underline{y}_{1,j} = [U_1]\underline{\alpha}_j \tag{6.44}$$

Equation (6.44) does not take advantage of the multiscale structure afforded by the wavelet basis. Recall that a multilevel expansion of the first-order kernel can be written as

$$h_{1,j}(\xi) = \sum_p \alpha_{j_0,p}\chi_{j_0,p}(\xi) + \sum_{l=j_0}^{j-1}\sum_p \beta_{l,p}\psi_{l,p}(\xi) \tag{6.45}$$

This multiscale representation is in terms of Haar wavelets on levels j_0 through $j-1$ and the scaling functions on the coarsest level j_0 only. Recall that this representation

is equivalent to the single-scale expansion of the kernel given in equation (6.39). For convenience, a vector of multiscale coefficients is defined as

$$\underline{\beta}_1 := \left\{ \begin{array}{c} \underline{\alpha}_{j_0} \\ \underline{\beta}_{j_0} \\ \underline{\beta}_{j_0+1} \\ \vdots \\ \underline{\beta}_{j-2} \\ \underline{\beta}_{j-1} \end{array} \right\}$$

where $\underline{\alpha}_{j_0}$ denotes all scaling function coefficients on level j_0, $\underline{\beta}_{j_0}$ represents all wavelet coefficients on level j_0, and so forth. In Chapter 3, it was shown that the multilevel coefficients in $\underline{\beta}_1$ can be computed from the single-scale coefficients $\underline{\alpha}_j$ via a recursive, fast-filter operation. It was also demonstrated that this decomposition can be written in matrix form such that

$$\underline{\beta}_1 = [T_1]\underline{\alpha}_j \tag{6.46}$$

where $[T_1]$ is the matrix realization of the wavelet transform operation. The reconstruction of $\underline{\alpha}_j$ from $\underline{\beta}_1$ is then simply given by

$$\underline{\alpha}_j = [T_1]^{-1}\underline{\beta}_1 \tag{6.47}$$

Substituting into equation (6.44), we have an expression for the discrete first-order output in terms of the multiscale kernel coefficients:

$$\underline{y}_{1,j} = [U_1][T_1]^{-1}\underline{\beta}_1 \tag{6.48}$$

In many cases, the wavelet coefficients that correspond to some of the higher-resolution levels contribute very little to the structure of the kernel and can be neglected. Then, the N_1-dimensional multiscale vector $\underline{\beta}_1$ can be truncated, reducing the number of

wavelet coefficients that must be determined to $\hat{N}_1 < N_1$. Then, equation (6.49) reduces to

$$\underline{y}_{1,j} = [U_1][\hat{T}_1]^{-1}\underline{\hat{\beta}}_1 \tag{6.49}$$

where $\underline{\hat{\beta}}_1$ is the truncated vector of multiscale coefficients and $[\hat{T}_1]^{-1}$ represents the first $\hat{N}_1$ columns of the reconstruction matrix $[T_1]^{-1}$.

In deriving a discrete form of the second-order Volterra operator, the same discretization is used for the input and output. Therefore, equation (6.29) yields N equations for the discrete second-order outputs:

$$y_{2,j}(t_n) = \int_0^{t_n}\int_0^{\eta} h_{2,tri}(\xi,\eta)u_j(t_n-\xi)u_j(t_n-\eta)d\xi d\eta, \qquad n = 1,\ldots,N \tag{6.50}$$

where $t_n = 2^{-j}n$. The zero order hold approximation of the input takes the same form as before:

$$\begin{aligned} u_j(t_n-\xi) &= \sum_{k=0}^{n-1} u_{j,k}\chi_{j,n-k-1}(\xi) \\ &= \sum_{k=0}^{n-1} u_{j,n-k-1}\chi_{j,k}(\xi) \end{aligned} \tag{6.51}$$

$$\begin{aligned} u_j(t_n-\eta) &= \sum_{m=0}^{n-1} u_{j,m}\chi_{j,n-m-1}(\eta) \\ &= \sum_{m=0}^{n-1} u_{j,n-m-1}\chi_{j,m}(\eta) \end{aligned} \tag{6.52}$$

where

$$u_{j,k} = 2^{-j/2}u(2^{-j}k) \tag{6.53}$$

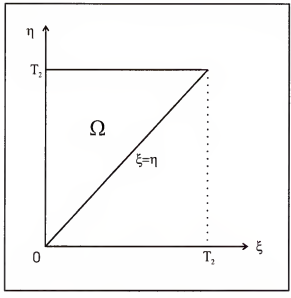


Figure 6.1: Triangular Domain of Support for the Second-Order Volterra Kernel

$$u_{j,m} = 2^{-j/2} u(2^{-j}m) \tag{6.54}$$

Defining the two-dimensional characteristic function

$$\chi_{j,(k,m)}(\xi,\eta) := \chi_{j,k}(\xi)\chi_{j,m}(\eta) \tag{6.55}$$

we have

$$u_j(t_n-\xi)u_j(t_n-\eta) = \sum_{k=0}^{n-1}\sum_{m=0}^{n-1} u_{j,n-k-1}u_{j,n-m-1}\chi_{j,(k,m)}(\xi,\eta) \tag{6.56}$$

Substituting into equation (6.50), we obtain

$$y_{2,j}(t_n) = \sum_{k=0}^{n-1}\sum_{m=0}^{n-1} u_{j,n-k-1}u_{j,n-m-1}\int_0^{t_n}\int_0^{\eta} h_{2,tri}(\xi,\eta)\chi_{j,(k,m)}(\xi,\eta)d\xi d\eta \tag{6.57}$$

Next, the triangular second-order kernel $h_{2,tri}$ is approximated in terms of the triangular wavelets derived in Chapter 5. The second-order kernel is discretized on level j_2 which, due to computational considerations, is typically a coarser resolution

level than level j. A single-scale expansion of the triangular kernel on level j_2 can be written as

$$h_{2,tri,j_2}(\xi,\eta) = \sum_{\mathbb{P}} \alpha_{j_2,\mathbb{P}} \phi_{j_2,\mathbb{P}}(\xi,\eta) \tag{6.58}$$

where $\{\phi_{j_2,\mathbb{P}}\}$ are triangular scaling functions. Recall that the notation $\mathbb{P}$ represents a j_2-dimensional multi-index $p_1 \ldots p_{j_2}$ where $p_i \in \{0,1,2,3\}$, $i = 1,\ldots,j_2$. The total number of coefficients $\{\alpha_{j_2,\mathbb{P}}\}$ in equation (6.58) is $N_2 = 4^{j_2}$. Once again, it is assumed that the system has finite memory and the second-order kernel can be truncated at some time $T_2 < t_N$. Then, the kernel is supported over the domain Ω shown in Figure (6.2). Substituting the single-scale approximation of the triangular kernel into equation (6.57), we obtain

$$y_{2,j}(t_n) = \sum_{k=0}^{n-1} \sum_{m=0}^{n-1} \sum_{\mathbb{P}} u_{j,n-k-1} u_{j,n-m-1} \alpha_{j_2,\mathbb{P}} \int_{\Omega} \chi_{j,(k,m)}(\xi,\eta) \phi_{j_2,\mathbb{P}}(\xi,\eta) d\xi d\eta \tag{6.59}$$

The truncation of the second-order kernel at time T_2 effectively lowers the upper summation limits on k and m to $\hat{N}-1$, where $\hat{N} = 2^{-j}T_2$. To be precise, the upper limits of these summations are $n-1$ for $n \leq \hat{N}$ and $\hat{N}-1$ for $n > \hat{N}$. Furthermore, the summations over k and m involve two-dimensional characteristic functions over the $[0,T_2]$ x $[0,T_2]$ square domain. When $m < k$, the function $\chi_{j,(k,m)}$ is not in the domain Ω. Hence, the integral in equation (6.59) is always zero when $m < k$ and the summation limits on m can be changed to $m = k$ to $\hat{N}-1$:

$$y_{2,j}(t_n) = \sum_{k=0}^{\hat{N}-1} \sum_{m=k}^{\hat{N}-1} \sum_{\mathbb{P}} u_{j,n-k-1} u_{j,n-m-1} \alpha_{j_2,\mathbb{P}} \int_{\Omega} \chi_{j,(k,m)}(\xi,\eta) \phi_{j_2,\mathbb{P}}(\xi,\eta) d\xi d\eta \tag{6.60}$$

These N equations for the discrete second-order outputs can be written in matrix form as

$$\underline{y}_{2,j} = [U_2][P]\underline{\alpha}_{tri,j_2} \tag{6.61}$$

where $\underline{y}_{2,j}$ is a vector of discrete second-order outputs and $\underline{\alpha}_{tri,j_2}$ is a vector of single-scale kernel coefficients. The quadratic input matrix $[U_2]$ takes the form

$$[U_2] = \begin{bmatrix} u_{j,0}^2 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ u_{j,1}^2 & u_{j,0}u_{j,1} & u_{j,0}^2 & 0 & 0 & 0 & 0 & \cdots & 0 \\ u_{j,2}^2 & u_{j,1}u_{j,2} & u_{j,1}^2 & u_{j,0}u_{j,2} & u_{j,0}u_{j,1} & u_{j,0}^2 & 0 & \cdots & 0 \\ \vdots & & & & & & & & \vdots \\ u_{j,\hat{N}-1}^2 & u_{j,\hat{N}-2}u_{j,\hat{N}-1} & u_{j,\hat{N}-2}^2 & \cdots & & & & \cdots & u_{j,0}^2 \\ u_{j,\hat{N}}^2 & u_{j,\hat{N}-1}u_{j,\hat{N}} & u_{j,\hat{N}-1}^2 & \cdots & & & & \cdots & u_{j,1}^2 \\ \vdots & & & & & & & & \vdots \\ u_{j,N-1}^2 & u_{j,N-2}u_{j,N-1} & u_{j,N-2}^2 & \cdots & & & & \cdots & u_{j,N-\hat{N}}^2 \end{bmatrix}$$

Clearly, $[U_2]$ has N rows. Each column in $[U_2]$ corresponds to a different characteristic function in Ω. There are $\frac{1}{2}(\hat{N}^2 + \hat{N})$ such functions in Ω, so there are $\frac{1}{2}(\hat{N}^2 + \hat{N})$ columns in $[U_2]$. The entries of the matrix $[P]$ correspond to the values of the integrals

$$\int_{\Omega} \chi_{j,(k,m)}(\xi,\eta)\phi_{j_2,\mathbb{p}}(\xi,\eta)d\xi d\eta \tag{6.62}$$

Each column in $[P]$ corresponds to a different triangular scaling function $\phi_{j_2,\mathbb{p}}$ while each row corresponds to a different characteristic function $\chi_{j,(k,m)}$. Therefore, $[P]$ is a $\frac{1}{2}(\hat{N}^2 + \hat{N})$ x 4^{j_2} matrix. Because each scaling function and characteristic function are supported over a relatively small subregion of Ω, most of the entries in $[P]$ are zero. Therefore, in practice, it is not efficient to calculate $[P]$ explicitly. Instead, the entries of the n x 4^{j_2} matrix product $[U_2][P]$ are calculated directly using equation (6.60).

The formulation in equation (6.61) does not exploit the multiscale structure of the discrete wavelet transform of $h_{2,tri}$. Recall that the single-scale approximation h_{2,tri,j_2} can be written in an equivalent, multilevel form

$$h_{2,tri,j_2} = \sum_{l=0}^{j_2-1} \sum_{\mathbb{P}} \sum_{r=1}^{3} \beta_{l,\mathbb{P}}^{r} \psi_{l,\mathbb{P}}^{r}(\xi,\eta) + \alpha\phi(\xi,\eta) \tag{6.63}$$

This representation is in terms of the wavelets on levels 0 through $j_2 - 1$ and the scaling function on level 0. The summation over r reflects the fact that there are three different types of triangular wavelets. By construction, the coarsest-resolution level is level 0, where the scaling function ϕ is simply a constant function over Ω. As always, the multilevel wavelet coefficients in equation (6.63) can be calculated from the single-scale coefficients in a fast, efficient manner. Defining a multiscale vector of coefficients

$$\underline{\beta}_2 := \begin{Bmatrix} \alpha \\ \underline{\beta}_0 \\ \underline{\beta}_1 \\ \vdots \\ \underline{\beta}_{j_2-2} \\ \underline{\beta}_{j_2-1} \end{Bmatrix}$$

where each $\underline{\beta}_l$, $l = 0, \ldots, j_2 - 1$, represents all the wavelet coefficients on level l, we can write

$$\underline{\beta}_2 = [T_2]\underline{\alpha}_{2,tri} \tag{6.64}$$

In equation (6.64), $[T_2]$ represents a decomposition matrix that performs the wavelet transform. Then, we have

$$\underline{\alpha}_{2,tri} = [T_2]^{-1}\underline{\beta}_2 \tag{6.65}$$

where $[T_2]^{-1}$ is the reconstruction matrix. Substituting for $\underline{\alpha}_{2,tri}$ in equation (6.61), we obtain

$$\underline{y}_{2,j} = [U_2][P][T_2]^{-1}\underline{\beta}_2 \tag{6.66}$$

Just as for the first-order kernel, in many cases the wavelet coefficients in $\underline{\beta}_2$ corresponding to some of the finer-resolution levels do not contribute much to the overall structure of the kernel. Then, $\underline{\beta}_2$ can be truncated to $\hat{N}_2$ nonzero coefficients and equation (6.66) becomes

$$\underline{y}_{2,j} = [U_2][P][\hat{T}_2]^{-1}\underline{\hat{\beta}}_2 \tag{6.67}$$

where $\underline{\hat{\beta}}_2$ is the truncated vector of wavelet coefficients and $[\hat{T}_2]^{-1}$ represents the first $\hat{N}_2$ columns of $[T_2]^{-1}$.

Given the models for the discrete outputs of the first and second-order Volterra operators:

$$\underline{y}_{1,j} = [U_1][\hat{T}_1]^{-1}\underline{\hat{\beta}}_1 = [A_1]\underline{\hat{\beta}}_1 \tag{6.68}$$

$$\underline{y}_{2,j} = [U_2][P][\hat{T}_2]^{-1}\underline{\hat{\beta}}_2 = [A_2]\underline{\hat{\beta}}_2 \tag{6.69}$$

the total output can be written as

$$\underline{y}_j = \underline{y}_{1,j} + \underline{y}_{2,j} \tag{6.70}$$

where $\underline{y}_j$ is a vector of sampled outputs. Then, the total model takes the form

$$\underline{y}_j = [A_1\ A_2]\begin{Bmatrix} \underline{\hat{\beta}}_1 \\ \underline{\hat{\beta}}_2 \end{Bmatrix} = [A]\underline{\hat{\beta}} \tag{6.71}$$

Thus, the kernel identification procedure reduces to a least-squares problem in which equation (6.71) must be solved for the wavelet coefficients in $\underline{\hat{\vartheta}}$ that represent the first and second-order Volterra kernels. It should be emphasized that the first and second-order kernels must be identified simultaneously because the Volterra operators are not orthogonal.

6.3 Multiwavelet Kernel Identification

In this section, a kernel identification algorithm is developed in which the first and second-order kernels are approximated in terms of the piecewise-polynomial multiwavelets derived in Chapter 4. Once again, the output of the first-order Volterra operator takes the form

$$y_1(t) = \int_0^t h_1(\xi)u(t-\xi)d\xi \tag{6.72}$$

In this case, however, the symmetric form of the second-order kernel, $h_{2,sym}$, is used instead of the triangular form. Then, the second-order Volterra operator can be written as

$$y_2(t) = \int_0^t \int_0^t h_{2,sym}(\xi,\eta)u(t-\xi)u(t-\eta)d\xi d\eta \tag{6.73}$$

where the symmetric second-order kernel $h_{2,sym}$ is defined over a square domain. It is demonstrated later that the symmetric kernel can be represented using two-dimensional tensor product multiwavelets that have been adapted to the square domain.

First, the input and output is discretized in the same manner as in the last section. A zero-order hold representation of the input is written as

$$u_j(\xi) = \sum_{k=0}^{N-1} u_{j,k}\chi_{j,k}(\xi) \tag{6.74}$$

where the coefficients $\{u_{j,k}\}$ are given by

$$u_{j,k} = 2^{-j/2} u(2^{-j}k), \qquad k = 0, \ldots, N-1 \tag{6.75}$$

and N is the total number of samples. As before, we have N equations for the discrete first-order outputs:

$$y_{1,j}(t_n) = \int_0^{t_n} h_1(\xi) u_j(t_n - \xi) d\xi \qquad n = 1, \ldots, N \tag{6.76}$$

where $t_n = 2^{-j}n$. Last section, the following expression was derived for the input $u_j(t_n - \xi)$:

$$\begin{aligned} u_j(t_n - \xi) &= \sum_{k=0}^{n-1} u_{j,k} \chi_{j,n-k-1}(\xi) \\ &= \sum_{k=0}^{n-1} u_{j,n-k-1} \chi_{j,k}(\xi) \end{aligned} \tag{6.77}$$

Next, the first-order kernel h_1 is approximated in terms of the piecewise-polynomial multiwavelets. A single-scale representation of the kernel on resolution level j_1 is given by

$$h_{1,j_1}(\xi) = \sum_p \sum_{s=1}^{r} \alpha_{j,p}^s \phi_{j,p}^s(\xi) \tag{6.78}$$

where r denotes the number of generators. Recall, from Chapter 4, that r depends on the order of the multiwavelets. For example, there are four piecewise-linear scaling functions, four piecewise-quadratic scaling functions, and five piecewise-cubic scaling functions. For computational purposes, the kernel is often discretized on a coarser-resolution level than the input and output. Once again, it is assumed that the system has finite memory and the first-order kernel can be truncated at time T_1. The kernel is then supported over the domain $[0, T_1]$ and the scaling functions in equation (6.78) are adapted to this domain in the manner detailed in Section 4.6. This implies that

p is summed from 0 to $2^{-j_1}T_1$, with an additional scaling function $\phi^1_{j_1,2^{j_1}}$ that is restricted to $[0,T_1]$. The scaling function $\phi^1_{j_1,0}$ is also restricted to $[0,T_1]$. Hence there are a total of $N_1 = 2^j T_1 r + 1$ scaling functions in the single-scale expansion of the first-order kernel.

Substituting equations (6.77) and (6.78) into equation (6.76), we obtain

$$y_{1,j}(t_n) = \sum_{k=0}^{\hat{N}-1} \sum_{p} \sum_{s=1}^{r} u_{j,n-k-1}\alpha_{j_1,p} \int_0^{T_1} \phi^s_{j_1,p}(\xi)\chi_{j,k}(\xi)d\xi \tag{6.79}$$

Note that, due to the fact that the kernel has been truncated, the sum over k has an upper limit of $\hat{N}-1$, where $\hat{N} = 2^j T_1$, and the upper integration limit is reduced to T_1. More precisely, the upper limit in the k summation is $n-1$ for $n < \hat{N}$ and $\hat{N}-1$ for $n \geq \hat{N}$. Equation (6.79) can be written in matrix form as

$$\underline{y}_{1,j} = [U_1][P_1]\underline{\alpha}_{j_1} \tag{6.80}$$

where $\underline{y}_{1,j}$ is a vector of discrete first-order outputs and $\underline{\alpha}_{j_1}$ is a vector of scaling function coefficients that represent the first-order kernel. The N x $\hat{N}$ input matrix $[U_1]$ takes the form

$$[U_1] = \begin{bmatrix} u_{j,0} & 0 & 0 & \cdots & 0 \\ u_{j,1} & u_{j,0} & 0 & \cdots & 0 \\ \vdots & & & & \vdots \\ u_{j,\hat{N}-1} & u_{j,\hat{N}-2} & u_{j,\hat{N}-3} & \cdots & u_{j,0} \\ u_{j,\hat{N}} & u_{j,\hat{N}-1} & u_{j,\hat{N}-2} & \cdots & u_{j,1} \\ \vdots & & & & \vdots \\ u_{j,N-1} & u_{j,N-2} & u_{j,N-3} & \cdots & u_{j,N-\hat{N}} \end{bmatrix}$$

and $[P_1]$ is a $\hat{N}$ x N_1 matrix whose entries take the form

$$\int_0^{T_1} \phi_{j_1,p}^s(\xi)\chi_{j,k}(\xi)d\xi$$

Each row of $[P_1]$ corresponds to a different characteristic function $\chi_{j,k}$ while each column corresponds to a different scaling function $\phi_{j_1,p}^s$. It should be noted that the matrix $[P_1]$ is never explicitly formed in practice, but rather the entries of the N x N_1 matrix product $[U_1][P_1]$ are calculated using equation (6.79).

At this point, we take advantage of the multiscale structure of the discrete wavelet transform of the first-order kernel. The wavelet transform yields the following multilevel representation of the kernel:

$$h_{1,j_1}(\xi) = \sum_p \sum_{s=1}^{r} \alpha_{j_0,p}^s \phi_{j_0,p}^s(\xi) + \sum_{l=j_0}^{j_1-1} \sum_p \sum_{s=1}^{r} \beta_{l,p}^s \psi_{l,p}^s(\xi) \tag{6.81}$$

where j_0 is the coarsest level used in the representation. Defining a vector of multiscale coefficients

$$\underline{\beta}_1 := \left\{ \begin{array}{c} \underline{\alpha}_{j_0} \\ \underline{\beta}_{j_0} \\ \underline{\beta}_{j_0+1} \\ \vdots \\ \underline{\beta}_{j_1-2} \\ \underline{\beta}_{j_1-1} \end{array} \right\}$$

we can write

$$\underline{\beta}_1 = [T_1]\underline{\alpha}_{j_1} \tag{6.82}$$

where $[T_1]$ is the decomposition matrix that performs the multiwavelet transform. Then,

$$\underline{\alpha}_{j_1} = [T_1]^{-1}\underline{\beta}_1 \tag{6.83}$$

and equation (6.80) takes the form

$$\underline{y}_{1,j} = [U_1][P_1][T_1]^{-1}\underline{\beta}_1 \tag{6.84}$$

As before, it can often be assumed that the wavelet coefficients corresponding to some of the finer-resolution levels in $\underline{\beta}_1$ can be neglected. Then, $\underline{\beta}_1$ can be truncated to $\hat{N}_1 < N_1$ nonzero terms. This results in the reduced-order model

$$\underline{y}_{1,j} = [U_1][P_1][\hat{T}_1]^{-1}\underline{\hat{\beta}}_1 \tag{6.85}$$

where $\underline{\hat{\beta}}_1$ is the truncated vector of wavelet coefficients and $[\hat{T}_1]^{-1}$ represents the first $\hat{N}_1$ columns of $[T_1]^{-1}$.

Using the same discretization of the input and output, the discrete second-order outputs can be written as

$$y_{2,j}(t_n) = \int_0^{t_n}\int_0^{t_n} h_{2,sym}(\xi,\eta)u_j(t_n-\xi)u_j(t_n-\eta)d\xi d\eta \qquad n = 1,\ldots,N \tag{6.86}$$

In the last section, it was shown that the zero-order hold approximation of the input yields the expression

$$u_j(t_n-\xi)u_j(t_n-\eta) = \sum_{k=0}^{n-1}\sum_{m=0}^{n-1} u_{j,n-k-1}u_{j,n-m-1}\chi_{j,(k,m)}(\xi,\eta) \tag{6.87}$$

where

$$u_{j,n-k-1} = u\left(2^{-j}(n-k-1)\right) \tag{6.88}$$

$$u_{j,n-m-1} = u\left(2^{-j}(n-m-1)\right) \tag{6.89}$$

Next, the second-order kernel is expanded in terms of the two-dimensional tensor product multiwavelets. Tensor product multiwavelets were discussed in detail in Section 3.6. A single-scale approximation of the second-order kernel on resolution level j_2 takes the form

$$\begin{aligned} h_{2,j_2}(\xi,\eta) &= \sum_p \sum_q \sum_{s=1}^{r} \sum_{v=1}^{r} \alpha^{(s,v)}_{j_2,\langle p,q\rangle} \phi^s_{j_2,p}(\xi)\phi^v_{j_2,q}(\eta) \\ &= \sum_p \sum_q \sum_{s=1}^{r} \sum_{v=1}^{r} \alpha^{(s,v)}_{j_2,\langle p,q\rangle} \Phi^{(s,v)}_{j_2,\langle p,q\rangle}(\xi,\eta) \end{aligned} \tag{6.90}$$

Once again, it is assumed that the system has finite memory and the kernel can be truncated at some time T_2. The two-dimensional scaling functions in equation (6.90) are adapted to the $[0,T_2]$ x $[0,T_2]$ square domain in the manner detailed in Section 4.6. Then, there are a total of $N_2 = (2^{j_2}T_2 r + 1)^2$ scaling functions in the single-scale expansion of the second-order kernel.

Substituting equations (6.87) and (6.90) into equation (6.86), we obtain the following expression for the discrete second-order outputs:

$$y_{2,j}(t_n) = \sum_{k,m=0}^{\tilde{N}-1} \sum_{p,q} \sum_{s,v=1}^{r} u_{j,n-k-1} u_{j,n-m-1} \alpha^{(s,v)}_{j_2,\langle p,q\rangle} \int_0^{T_2}\int_0^{T_2} \Phi^{(s,v)}_{j_2,\langle p,q\rangle}(\xi,\eta)\chi_{j,(k,m)}(\xi,\eta)d\xi d\eta \tag{6.91}$$

Because the second-order kernel has been truncated at time T_2, the upper limits in the k and m summations have been lowered to $\tilde{N}-1$, where $\tilde{N} = 2^j T_2$, and the upper integration limits have been changed to T_2. More specifically, the upper summation limits for k and m are $n-1$ for $n < \tilde{N}$ and $\tilde{N}-1$ for $n \geq \tilde{N}$. Equation (6.91) can be written in matrix form:

$$\underline{y}_{2,j} = [U_2][P_2]\underline{\alpha}_{j_2} \tag{6.92}$$

where $\underline{y}_{2,j}$ is a vector of discrete second-order outputs and $\underline{\alpha}_{j_2}$ is a vector of two-dimensional scaling function coefficients. The quadratic input matrix $[U_2]$ takes the form

$$[U_2]^T = \begin{bmatrix}
u_{j,0}^2 & u_{j,1}^2 & \cdots & u_{j,\tilde{N}-1}^2 & u_{j,\tilde{N}}^2 & \cdots & u_{j,N-1}^2 \\
0 & u_{j,1}u_{j,0} & \cdots & u_{j,\tilde{N}-1}u_{j,\tilde{N}-2} & u_{j,\tilde{N}}u_{j,\tilde{N}-1} & \cdots & u_{j,N-1}u_{j,N-2} \\
0 & 0 & \cdots & u_{j,\tilde{N}-1}u_{j,\tilde{N}-3} & u_{j,\tilde{N}}u_{j,\tilde{N}-2} & \cdots & u_{j,N-1}u_{j,N-3} \\
\vdots & \vdots & & \vdots & \vdots & & \vdots \\
0 & 0 & \cdots & u_{j,\tilde{N}-1}u_{j,0} & u_{j,\tilde{N}}u_{j,1} & \cdots & u_{j,N-1}u_{j,N-\tilde{N}} \\
0 & u_{j,0}u_{j,1} & \cdots & u_{j,\tilde{N}-2}u_{j,\tilde{N}-1} & u_{j,\tilde{N}-1}u_{j,\tilde{N}} & \cdots & u_{j,N-2}u_{j,N-1} \\
0 & u_{j,0}^2 & \cdots & u_{j,\tilde{N}-2}^2 & u_{j,\tilde{N}-1}^2 & \cdots & u_{j,N-2}^2 \\
0 & 0 & \cdots & u_{j,\tilde{N}-2}u_{j,\tilde{N}-3} & u_{j,\tilde{N}-1}u_{j,\tilde{N}-2} & \cdots & u_{j,N-2}u_{j,N-3} \\
\vdots & \vdots & & \vdots & \vdots & & \vdots \\
0 & 0 & \cdots & u_{j,\tilde{N}-2}u_{j,0} & u_{j,\tilde{N}-1}u_{j,1} & \cdots & u_{j,N-2}u_{j,N-\tilde{N}} \\
\vdots & \vdots & & \vdots & \vdots & & \vdots \\
0 & 0 & \cdots & u_{j,0}u_{j,\tilde{N}-1} & u_{j,1}u_{j,\tilde{N}} & \cdots & u_{j,N-\tilde{N}}u_{j,N-1} \\
\vdots & \vdots & & \vdots & \vdots & & \vdots \\
0 & 0 & \cdots & u_{j,0}^2 & u_{j,1}^2 & \cdots & u_{j,N-\tilde{N}}^2
\end{bmatrix}$$

There are a total of N rows in $[U_2]$. Each column of $[U_2]$ corresponds to a different two-dimensional characteristic function $\chi_{j,(k,m)}$ in the $[0,T_2]$ x $[0,T_2]$ domain. There are a total of $\tilde{N}^2$ such functions; therefore, $[U_2]$ has $\tilde{N}^2$ columns. The matrix $[P_2]$ is composed of integrals of the form

$$\int_0^{T_2}\int_0^{T_2} \Phi_{j_2,(p,q)}^{(s,v)}(\xi,\eta)\chi_{j,(k,m)}(\xi,\eta)d\xi d\eta \tag{6.93}$$

where each row corresponds to a different characteristic function $\chi_{j,(k,m)}$ and each column corresponds to a different two-dimensional scaling function $\Phi^{(s,v)}_{j_2,(p,q)}$. Therefore, $[P_2]$ is an $\tilde{N}^2$ x N_2 matrix.

As a final step, recall that the single-scale expansion of the second-order kernel can be written equivalently as a multiscale representation:

$$\begin{aligned}
h_{2,j_2}(\xi,\eta) &= \sum_{p,q}\sum_{s=1}^{r}\sum_{v=1}^{r} \alpha^{(s,v)}_{j_0,(p,q)}\phi^{s}_{j_0,p}(\xi)\phi^{v}_{j_0,q}(\eta) + \sum_{l=j_0}^{j_2-1}\sum_{p,q}\sum_{s=1}^{r}\sum_{v=1}^{r} \beta^{1,(s,v)}_{l,(p,q)}\phi^{s}_{l,p}(\xi)\psi^{v}_{l,q}(\eta) \\
&+ \sum_{l=j_0}^{j_2-1}\sum_{p,q}\sum_{s=1}^{r}\sum_{v=1}^{r} \beta^{1,(s,v)}_{2,(p,q)}\psi^{s}_{l,p}(\xi)\phi^{v}_{l,q}(\eta) + \sum_{l=j_0}^{j_2-1}\sum_{p,q}\sum_{s=1}^{r}\sum_{v=1}^{r} \beta^{1,(s,v)}_{3,(p,q)}\psi^{s}_{l,p}(\xi)\psi^{v}_{l,q}(\eta) \\
&= \sum_{p,q}\sum_{s=1}^{r}\sum_{v=1}^{r} \alpha^{(s,v)}_{j_0,(p,q)}\Phi^{(s,v)}_{j_0,(p,q)}(\xi,\eta) + \sum_{l=j_0}^{j_2-1}\sum_{p,q}\sum_{s=1}^{r}\sum_{v=1}^{r} \beta^{1,(s,v)}_{l,(p,q)}\Psi^{1,(s,v)}_{l,(p,q)}(\xi,\eta) \\
&+ \sum_{l=j_0}^{j_2-1}\sum_{p,q}\sum_{s=1}^{r}\sum_{v=1}^{r} \beta^{2,(s,v)}_{l,(p,q)}\Psi^{2,(s,v)}_{l,(p,q)}(\xi,\eta) + \sum_{l=j_0}^{j_2-1}\sum_{p,q}\sum_{s=1}^{r}\sum_{v=1}^{r} \beta^{3,(s,v)}_{l,(p,q)}\Psi^{3,(s,v)}_{l,(p,q)}(\xi,\eta)
\end{aligned} \tag{6.94}$$

which is in terms of the three different types of tensor product wavelets $\{\Psi^1,\Psi^2,\Psi^3\}$ on levels j_0 through j_2-1 and the scaling functions on the coarsest level j_0. In Section 3.6, it was demonstrated that there are fast algorithms whereby the coefficients in this multiscale expansion can be calculated from the single-scale coefficients. Defining a vector of multiscale coefficients

$$\underline{\beta}_2 := \left\{ \begin{array}{c} \underline{\alpha}_{j_0} \\ \underline{\beta}_{j_0} \\ \underline{\beta}_{j_0+1} \\ \vdots \\ \underline{\beta}_{j_2-2} \\ \underline{\beta}_{j_2-1} \end{array} \right\}$$

in which all the coefficients corresponding to a particular level have been grouped together in vector form, we can write

$$\underline{\beta}_2 = [T_2]\underline{\alpha}_{j2} \tag{6.95}$$

where $[T_2]$ is a decomposition matrix that performs the two-dimensional wavelet transform. Similarly,

$$\underline{\alpha}_{j2} = [T_2]^{-1}\underline{\beta}_2 \tag{6.96}$$

where $[T_2]^{-1}$ is the reconstruction matrix. Substituting into equation (6.92), we obtain

$$\underline{y}_{2,j} = [U_2][P_2][T_2]^{-1}\underline{\beta}_2 \tag{6.97}$$

Assuming that the wavelet coefficients corresponding to some of the finer-resolution levels can be neglected, the multiscale vector $\underline{\beta}_2$ can be truncated to $\hat{N}_2$ nonzero terms. Then, we obtain the following model for the second-order output:

$$\underline{y}_{2,j} = [U_2][P_2][\hat{T}_2]^{-1}\hat{\underline{\beta}}_2 \tag{6.98}$$

where $\hat{\underline{\beta}}_2$ is the truncated vector of multiscale coefficients and $[\hat{T}_2]^{-1}$ represents the first $\hat{N}_2$ columns of $[T_2]^{-1}$.

Given the models for the discrete outputs of the first and second-order Volterra operators:

$$\underline{y}_{1,j} = [U_1][P_1][\hat{T}_1]^{-1}\hat{\underline{\beta}}_1 = [A_1]\hat{\underline{\beta}}_1 \tag{6.99}$$

$$\underline{y}_{2,j} = [U_2][P_2][\hat{T}_2]^{-1}\hat{\underline{\beta}}_2 = [A_2]\hat{\underline{\beta}}_2 \tag{6.100}$$

the sampled output can be written as

$$\underline{y}_j = \underline{y}_{1,j} + \underline{y}_{2,j} \tag{6.101}$$

Then, we obtain a combined model of the form

$$\underline{y}_j = [A_1 \; A_2] \begin{Bmatrix} \hat{\underline{\beta}}_1 \\ \hat{\underline{\beta}}_2 \end{Bmatrix} = [A]\hat{\underline{\beta}} \tag{6.102}$$

where $\underline{y}_j$ is a vector of sampled outputs. Equation (6.102) can then be solved in a least-squares sense for the wavelet coefficients $\hat{\underline{\beta}}$ that represent the first and second-order kernels. Once again, it is necessary to identify the first and second-order kernels simultaneously because the terms in the Volterra series are not orthogonal. Recall, from Section 2.3, that the second-order kernel over a square domain can take several forms, where only the symmetric form is unique. Therefore, it is preferrable to have the identified second-order kernel h_{2,j_2} in symmetric form. This is easily accomplished via the relationship

$$h_{2,sym,j_2}(\xi,\eta) = \tfrac{1}{2}\{h_{2,j_2}(\xi,\eta) + h_{2,j_2}(\eta,\xi)\} \tag{6.103}$$

The j_2 subscript is left on the terms in equation (6.103) to denote that the second-order kernel has been approximated on resolution level j_2.

6.4 The Least-Squares Problem

In the previous two sections, it has been shown that both wavelet-based kernel identification algorithms can be reduced to a least-squares problem that must be solved for the unknown wavelet coefficients. That is, the problem can be formulated as

$$[A]\hat{\underline{\beta}} = \underline{y} \tag{6.104}$$

where the matrix $[A]$ is known, $\underline{y}$ is a vector of measured outputs, and $\underline{\hat{\beta}}$ is a vector of unknown wavelet coefficients that represents the first and second-order Volterra kernels. In this discussion, it is assumed that $[A]$ is an m x n matrix, $\underline{y}$ is an m-dimensional vector, and $\underline{\hat{\beta}}$ is an n-dimensional vector. Furthermore, it is assumed that the system is overdetermined ($m > n$) since, in general, a large number of observations yields a more accurate least-squares solution.

Given a solution $\underline{\hat{\beta}}$ to equation (6.104), the residual $\underline{\nu}$ is defined as

$$\underline{\nu} := \underline{y} - [A]\underline{\hat{\beta}} \tag{6.105}$$

Then, a least-squares solution $\underline{\hat{\beta}}^*$ can be found that minimizes the least-square error

$$e = \|\underline{\nu}\| = \sqrt{\underline{\nu}^T \underline{\nu}} \tag{6.106}$$

In other words, the least-squares solution $\underline{\hat{\beta}}^*$ satisfies

$$\|\underline{y} - [A]\underline{\hat{\beta}}^*\| \leq \|\underline{y} - [A]\underline{\hat{\beta}}\| \tag{6.107}$$

for all n-dimensional vectors $\underline{\hat{\beta}}$ [111]. The estimated solution $[A]\underline{\hat{\beta}}^*$ is a weighted combination of the columns of $[A]$ and is therefore in the column space of $[A]$, the space spanned by the columns of $[A]$. This space, denoted $Col(A)$, is a subset of the space $\mathbb{R}^m$ that contains all m-dimensional vectors. That is, $Col(A) \subseteq \mathbb{R}^m$. Thus, the least-squares solution $\underline{\hat{\beta}}^*$ is the value of $\underline{\hat{\beta}}$ that minimizes the difference between $\underline{y} \in \mathbb{R}^m$ and $[A]\underline{\hat{\beta}} \in Col(A)$. It is a well-known theorem from linear algebra that the orthogonal projection of $\underline{y}$ onto $Col(A)$

$$\underline{y}^* = \mathrm{Proj}_{Col(A)}\underline{y} \tag{6.108}$$

minimizes this difference [111]. Hence, the least-squares solution $\underline{\hat{\beta}}^*$ satisfies

$$[A]\underline{\hat{\beta}}^* = \underline{y}^* \tag{6.109}$$

Since $\underline{y}^*$ is the orthogonal projection of $\underline{y}$ onto $Col(A)$, it follows that the difference $\underline{y} - \underline{y}^*$ is orthogonal to $Col(A)$. Therefore, each column of $[A]$ must be orthogonal to $\underline{y} - \underline{y}^*$:

$$\underline{a}_i^T(\underline{y} - \underline{y}^*) = \underline{a}_i^T(\underline{y} - [A]\underline{\hat{\beta}}^*) = 0, \qquad i = 1, \ldots, n$$

where $\underline{a}_i$ denotes the ith column of $[A]$. Then, we can write

$$[A]^T(\underline{y} - [A]\underline{\hat{\beta}}^*) = \underline{0}$$

which leads to the standard normal equation

$$[A]^T[A]\underline{\hat{\beta}}^* = [A]^T\underline{y} \tag{6.110}$$

which can be solved for the least-squares solution $\underline{\hat{\beta}}^*$. For the case where $[A]$ has full rank, or equivalently in the overdetermined case, $Col(A) = \mathbb{R}^m$, equation (6.110) has a unique solution given by

$$\underline{\hat{\beta}}^* = \left([A]^T[A]\right)^{-1}[A]^T\underline{y} \tag{6.111}$$

When $[A]$ is an ill-conditioned matrix, meaning that small errors in the entries of $[A]$ can result in large errors in the computed solution, factorization techniques such as QR factorization, Cholesky factorization, and the singular value decompsition (SVD) can sometimes be used to generate an accurate solution to the least-squares problem.

The identification of Volterra kernels falls into the category of inverse problems. That is, given input/output data from the system, the task is to identify the Volterra kernels that characterize the system. This is in contrast to direct problems where, given a model of the system and an input, the goal is to predict the system response.

An unfortunate characteristic of inverse problems is that they are frequently ill-posed in the sense that small errors in the matrix $[A]$ and the output measurement vector $\underline{y}$ can lead to unstable solutions. Hadamard is often credited with formally defining a well-posed problem to be one for which a solution exists, the solution is unique, and the solution depends continuously on the data [112]. The last requirement specifies that small errors in the data should lead to only small errors in the solution. An ill-posed problem, then, can be defined as any problem that is not well-posed. There is a great deal of literature pertaining to ill-posed problems, including books by Groetsch [113] and Hansen [114]. The matrix formulations of the wavelet-based kernel identification algorithms always include small errors due to the zero-order hold discretization of the input and output. Furthermore, when working with experimental data, the measured output always contains some degree of noise. Because of these errors and the ill-posed nature of the problem, a least-squares solution of equation (6.104) based on conventional solution techniques is often very unstable. For this reason, regularization methods such as Tikhonov regularization and the truncated singular value decomposition are needed in order to generate stable solutions to these ill-posed problems. Tikhonov regularization effectively replaces an ill-posed problem with a family of nearby well-posed problems [113]. The normal equation, equation (6.110), is written in the regularized form

$$\left([A]^T[A] + \alpha[I]\right) \underline{\hat{\beta}}_\alpha = [A]^T \underline{y} \tag{6.112}$$

where α is a regularization parameter and $[I]$ is the identity matrix. Then, the least-squares solution $\underline{\hat{\beta}}_\alpha$ can be obtained as

$$\underline{\hat{\beta}}_\alpha = \left([A]^T[A] + \alpha[I]\right)^{-1} [A]^T \underline{y} \tag{6.113}$$

As the parameter α approaches 0, equation (6.112) approaches the original normal equation. Basically, α is chosen to be large enough so that equation (6.113) yields a stable solution.

The regularization technique used in this dissertation for solving the wavelet-based kernel identification problem is the truncated singular value decomposition. A discussion of the singular value decomposition can be found in any elementary linear algebra text [111], [115]. The singular value decomposition (SVD) of an m x n matrix $[A]$ is a factorization of the form

$$[A] = [U][\Sigma][V]^T \tag{6.114}$$

where $[U]$ and $[V]$ are orthogonal matrices. The m x n matrix $[\Sigma]$ can be partitioned into a diagonal block $[D]$ and blocks of zero as

$$[\Sigma] = \left[\begin{array}{c|c} [D]_{r \times r} & 0_{r \times (n-r)} \\ --- & --- \\ 0_{(m-r) \times r} & 0_{(m-r) \times (n-r)} \end{array}\right]$$

The diagonal matrix $[D]$ takes the form

$$[D] = \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_r \end{bmatrix}$$

where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ are the ordered singular values of $[A]$ and r is the rank of $[A]$. The singular values correspond to the square roots of the eigenvalues of the symmetric matrix $[A]^T[A]$. It is not difficult to prove that all eigenvalues of $[A]^T[A]$ are real and positive [111]. Therefore, the r singular values are also real and positive. Due to the structure of $[\Sigma]$, it is clear that the last $(m-r)$ columns of $[U]$ and the last

$(n - r)$ rows of $[V]^T$ do not contribute to $[A]$. Then, the SVD of $[A]$ can be written in the reduced form

$$[A] = [U_r][D][V_r]^T \tag{6.115}$$

where $[U_r]$ and $[V_r]$ denote the first r columns of $[U]$ and $[V]$ respectively. Equation (6.115) can then be used to define the pseudoinverse of $[A]$, given by

$$[A]^+ = [V_r][D]^{-1}[U_r]^T \tag{6.116}$$

where the fact that $[U]$ and $[V]$ are orthogonal matrices has been used. The matrix $[D]^{-1}$ is a diagonal matrix composed of the reciprocals of the singular values of $[A]$. It is not difficult to verify that the pseudoinverse $[A]^+$ acts as an inverse in the sense that

$$[A]^+[A] = [A][A]^+ = [I]$$

The pseudoinverse can be used to compute a least-squares solution to equation (6.104) of the form

$$\underline{\hat{\beta}}^* = [A]^+\underline{y} \tag{6.117}$$

It can be demonstrated that the solution given by equation (6.117) is indeed a least-squares solution [111]. The pseudoinverse can alternatively be expressed as

$$[A]^+ = \sum_{i=1}^{r} \frac{1}{\sigma_i} \underline{v}_i \underline{u}_i^T \tag{6.118}$$

where $\underline{u}_i$ and $\underline{v}_i$ represent the ith columns of $[U]$ and $[V]$. Then, the least-squares solution can be written as

$$\underline{\hat{\beta}}^* = \sum_{i=1}^{r} \frac{1}{\sigma_i} \left(\underline{u}_i^T \underline{y}\right) \underline{v}_i \tag{6.119}$$

Ill-posed problems are characterized by the fact that the singular values $\{\sigma_i\}$ approach zero as i approaches infinity. Therefore, equation (6.119) tends to yield unstable solutions due to the $\frac{1}{\sigma_i}$ factors. The truncated singular value decomposition is a regularization technique whereby the influence of some of the smaller singular values is removed by truncating the summation in equation (6.119) [113]:

$$\underline{\hat{\beta}}^* = \sum_{i=1}^{p} \frac{1}{\sigma_i} \left(\underline{u}_i^T \underline{y}\right) \underline{v}_i \tag{6.120}$$

The choice of truncation level p in equation (6.120) is not always a straightforward matter. If p is chosen to be too high (i.e., too many singular values are retained), the resulting solution will be unstable. Conversely, if too few singular values are kept, the solution will be inaccurate because too much information is lost and the regularized problem does not closely resemble the original problem. Determining an optimal cutoff point often requires viewing a plot of the singular values and trying different truncation points. Qualitatively, an effective cutoff point is often indicated by a leveling off of the singular values followed by a precipitious decline. In this case, just before the singular values decline often serves as a good truncation point. In their text, Colton and Kress [112] discuss the determination of the minimum number of singular values needed to ensure that the norm of the estimation error is less than a given value δ. A theorem is given that guarantees that such a truncation point exists for any choice of δ. In any event, the truncated singular value decomposition has proven to be an effective tool for obtaining stable solutions to the least-squares problem and extracting wavelet-based kernel approximations.

6.5 Implementation

Both wavelet-based kernel identification algorithms discussed in this chapter are implemented using MATLAB programs. In this section, some of the details of these implementations are discussed. Then, in the following chapter, several examples will be given.

The first step in both implementations is to load the input and output data. The algorithms require a dyadic sampling rate for the data; that is, the sampling rate must be 2^j Hz, $j \in \mathbb{Z}$. When the data is obtained via simulations, this requirement is easily satisfied. If however, the data is obtained experimentally, the sampling rate might not be dyadic. In this case, a separate MATLAB routine projects the data onto Haar scaling functions on a chosen level $j \in \mathbb{Z}$. This effectively generates a zero-order hold approximation of the input and output with a sampling step of 2^{-j}. The user sets several parameters for both kernel identification algorithms. The sampling level j must be provided, as well as the starting time (usually chosen to be 0) and final time for the input/output data that is to be used for the identification. The user also sets the finest-scale approximation levels for the first and second-order Volterra kernels. For the triangular wavelet implementation, the finest-scale level for the Haar representation of the first-order kernel is automatically chosen to be the same as the discretization level of the input and output. The discretization level for the second-order triangular kernel is also chosen. For the multiwavelet implementation, separate discretization levels are set for the first and second-order kernels. These levels are almost always selected to be coarser than the input/output sampling level due to computational considerations. In addition, the user chooses whether to use piecewise-linear, quadratic, or cubic multiwavelets to approximate the kernels. The assumption of finite memory implies that the kernels decay to zero in a finite period of time. Therefore, time durations must also be chosen for the first and second-order kernels. In several of the simulation examples given in the next chapter, the

kernels are known in analytical form. Therefore, the user knows *a priori* the effective time durations of these kernels. In the more general case where such knowledge is not available, the time durations are determined via trial and error. It is usually most effective to start with relatively large kernels. From the plots of the identified kernels, it can easily be determined if the kernels can be truncated at earlier times. Finally, the user must specify the coarsest level j_0 to be used in the multiscale wavelet representations of the kernels. Usually, this is chosen to be the coarsest possible level to give the largest number of different scales.

Once the parameters for the model have been set, the coefficient matrices $[A_1]$ and $[A_2]$ are formed. In the triangular implementation, the first-order input matrix $[U_1]$ is formed as described in Section 6.2. Reconstruction functions have been written in MATLAB for the Haar wavelet and the triangular wavelets. These routines peform the wavelet reconstruction via a recursive application of equation (3.89) for the Haar wavelet and equation (5.93) for the triangular wavelets. The reconstruction matrix $[T_1]^{-1}$ is then formed column by column by sending a column vector of zeros with a 1 in the ith row to the Haar reconstruction function. This effectively sifts out the ith column of the reconstruction matrix $[T_1]^{-1}$. The triangular reconstruction matrix $[T_2]^{-1}$ is constructed in a similar manner. In the second-order model, the matrix $[U_2][P_2]$ in equation (6.67) is formed together rather than forming $[U_2]$ and $[P_2]$ explicitly and multiplying them. This is done for computational efficiency since the matrix $[P_2]$ is sparse. In order to compute this matrix product, integrals of two-dimensional characteristic functions and triangular scaling functions of the form in equation (6.62) must be calculated. This is not a simple task because the indexing of the triangular scaling functions is rather complex. The calculation of the integrals is accomplished via a subroutine that loops over the scaling functions and determines

which characteristic functions overlap the given scaling function. Then, the corresponding inner products are calculated and equation (6.60) is used to generate the entry of $[U_2][P_2]$ corresponding to that particular scaling function.

In the multiwavelet implementation, both the first and second-order models require the computation of integrals of the products of scaling functions and characteristic functions. The limits on these integrations is dependent upon the difference in the resolution levels of the kernel and the input. Unlike the triangular implementation, where the integrals were of products of constant functions, the integrals in the multiwavelet implementation involve products of piecewise-polynomial scaling functions and characteristic functions. Therefore, a separate quadrature subroutine evaluates the one-dimensional integrals numerically using Gauss-Legendre quadrature. The resulting values of these integrals over one 2^{-j_1} block are stored in an array, where j_1 is the resolution level of the first-order kernel. In order to do this, the one scaling function that has support over $[-1, 1]$ is treated as two "half functions." It is only necessary to perform these integrations over one block since integrals over subsequent blocks have the same values. Then, the array of integral values is of dimension $(r + 1)$ x 2^{j-j_1}, where 2^{j-j_1} is the number of characteristic functions that intersect a single scaling function. Because the two-dimensional integrals involve tensor product scaling functions and characteristic functions, they are actually comprised of the product of two one-dimensional integrals. Therefore, the two-dimensional integrals are evaluated by first generating an $(r + 1)$ x 2^{j-j_2} array of one-dimensional integrals of scaling functions and characteristic functions, where j_2 is the resolution level of the second-order kernel. Then, the two-dimensional integrals are calculated by multiplying the appropriate two entries in the array of one-dimensional integral values. The entries of the matrix products $[U_1][P_1]$ and $[U_2][P_2]$ are then calculated using equations (6.79) and (6.91) respectively. Just as in the triangular case, reconstruction functions have been written for the multiwavelets in one and two dimensions. These

routines perform the wavelet reconstruction through a recursive application of equation (3.157) for the one-dimensional case and equation (3.184) in the two-dimensional case. Because the multiwavelets are adapted to fit the domains of support of the first and second-order kernels, slight modifications have been made to these equations to account for the boundary coefficients. The one-dimensional multiwavelet reconstruction matrix $[T_1]^{-1}$ is generated column by column using the same procedure as the Haar case. Unfortunately, the two-dimensional case is not as simple as there is no convenient matrix representation of the tensor product multiwavelet reconstruction. Hence, in practice, the matrix $[T_2]^{-1}$ is not calculated and the single-scale coefficients $\underline{\alpha}_{j_2}$ are left in the model.

In both implementations, once the first and second-order models are formed, they are combined into one matrix. The resulting matrix equation is then solved for the wavelet coefficients that represent the first and second-order kernels. As discussed in the previous section, the least-squares solution is obtained using the truncated singular value decomposition. The singular value decomposition is computed using the MATLAB "svd" function and the singular values are then plotted. The user then determines where to truncate the singular values and the pseudoinverse in equation (6.120) is calculated using the MATLAB "pinv" function. The pseudoinverse is used to solve for the wavelet coefficients. In order to plot the resulting kernels, the single-scale coefficients are reconstructed from the multiscale wavelet coefficients using the reconstruction functions. The kernels are then plotted in terms of the scaling functions using plotting routines that have been developed for the Haar scaling functions, triangular scaling functions, and multiwavelet scaling functions in one and two dimensions. The first and second-order outputs corresponding to the identified Volterra kernels are also calculated. These outputs, as well as the total predicted output, are plotted in comparison to the actual sampled output. The main advantage of using the multiscale wavelet coefficients in the model is that the coefficients

corresponding to certain finer-resolution levels can easily be neglected via truncation. Hence, the number of nonzero wavelet coefficients to be used in the representation of the first and second-order kernels is a user-defined parameter. Usually, it makes sense to generate the full-order model first. Then, the number of nonzero coefficients can be systematically reduced. This corresponds to eliminating certain columns in the full-order matrix. In this manner, one can determine the fewest number of wavelet coefficients needed to represent the first and second-order kernels. Of course, the approximations of the kernels should improve as the number of nonzero coefficients is increased; however, including more wavelet coefficients than is necessary can actually make it more difficult to generate a stable least-squares solution. As a final note regarding the two-dimensional multiwavelet coefficients, the effect of having the single-scale coefficients in the model instead of the multiscale coefficients is that the vector of coefficients can no longer be truncated to neglect certain levels in the approximation. The number of nonzero terms must be varied by changing the initial resolution level of the kernel. For example, if a multiscale representation is desired over levels j_0 through $j_0 + 2$ only, the resolution level for the single-scale coefficients should be set to $j_0 + 3$. The two representations are equivalent. Thus, by varying the initial resolution level of the kernel, the number of nonzero wavelet coefficients is also varied. The disadvantage of having to do this is that the second-order model must be regenerated every time as opposed to just truncating the original full-order model.

At this point, several comments should be made regarding the computational limitations of these algorithms. In both cases, the main limiting factor is the total number of unknown coefficients to be determined. There is a limit to the size of a given matrix for which MATLAB can generate a singular value decomposition. In general, this limit occurs at around 2000 coefficients. Of course, the number of input/output data points is also a factor. This computational limit sets a constraint

on the resolution levels that can be chosen for the first and second-order kernels. The truncation of the kernels is an important parameter since using shorter-duration kernels reduces both the number of coefficients in the model and the computational time to form the model matrix. This, in turn, enables the user to choose higher levels of resolution for the kernels. Therefore, it is always advantageous to use the shortest possible time durations for the kernels. It should also be noted that, for the triangular wavelets, the resolution of the triangular kernel is directly dependent on the size of the kernel as well as the chosen resolution level. This is due to the method of construction of the triangular wavelets. For example, a finest-resolution level of 4 yields a single-scale approximation of the triangular kernel in terms of $4^4 = 256$ scaling functions. Clearly, this number of coefficients provides better resolution for a kernel with a duration of 1 second than for one with a time duration of 8 seconds. In the multiwavelet implementation, the maximum resolution level is limited in part by the fact that there are multiple scaling functions and wavelets. Given r scaling functions and wavelets, a representation of the first-order kernel requires approximately r times the number of functions as a Haar approximation at the same level. This difference is offset, however, by the fact that the multiwavelets have higher approximation order than the Haar wavelet. Therefore, an accurate representation of the kernel can be achieved at a lower resolution level using the multiwavelets. In any event, the current computational limits of the MATLAB implementations put an effective limit on the size of the kernels that can be identified. In general, this limit is about 32 seconds for the first-order kernel and 8 seconds for the second-order kernel. Despite these limitations, as will be demonstrated in the next chapter, first and second-order kernels can be identified for a large number of systems. In many cases, accurate estimates of the kernels can be obtained with a relatively small number of wavelet coefficents.

CHAPTER 7
NUMERICAL RESULTS

In this chapter, the wavelet-based kernel identification algorithms developed in the last chapter are demonstrated on several systems. First, a damped linear oscillator is considered. This linear system can be completely characterized in terms of the first-order Volterra kernel which, for a linear system, is equivalent to the impulse response function. The wavelet-based algorithms are used to identify the first-order kernel of this system. Also, the effect of varying kernel length is investigated for this system. The next example is a quadratic nonlinear system whose dynamics are completely described in terms of the second-order kernel. This system serves as a good validation case for the second-order parts of the two kernel identification algorithms. Next, a nonlinear oscillator is analyzed. This system consists of a linear oscillator with an added quadratic nonlinearity. First and second-order kernels are identified for this system. The identified kernels are then validated by comparing the output predicted by the kernels to the simulated output of the system under a variety of inputs. As a final example, Volterra kernels are extracted from flight flutter data from the Aerostructures Test Wing, which was flight-tested at NASA Dryden in 2001. It is demonstrated that the wavelet-based algorithms can be used to identify kernels from noisy flight data. This example also provides a practical application of these kernel identification techniques as the identified first-order kernels can be used to obtain improved flutter estimates for aeroelastic systems.

7.1 Linear Oscillator

In this section, first-order Volterra kernels are identified for a damped linear oscillator. The equation of motion for this system is given by

$$m\ddot{y}(t) + c\dot{y}(t) + ky(t) = u(t) \tag{7.1}$$

where y is the output displacement and u is the input force. The system parameters are the mass m, the damping coefficient c, and the spring stiffness k. In this example, these parameters have been chosen as

$$\begin{aligned} m &= 2 \text{ kg} \\ c &= 3.2 \text{ N} \cdot \text{s/m} \\ k &= 32 \text{ N/m} \end{aligned} \tag{7.2}$$

The initial position $y(0)$ and the initial velocity $\dot{y}(0)$ are both chosen to be zero. These zero initial conditions ensure that the zero-order kernel h_0 is zero in this example. Equation (7.1) is commonly written in the form

$$\ddot{y}(t) + 2\zeta\omega_n\dot{y}(t) + \omega_n^2 y(t) = \frac{1}{m}u(t) \tag{7.3}$$

where the damping ratio ζ and the natural frequency ω_n are given by

$$\omega_n = \sqrt{\frac{k}{m}} \tag{7.4}$$

$$\zeta = \frac{c}{\sqrt{km}} \tag{7.5}$$

For the system parameters given in equation (7.2), we have

$$\begin{aligned} \omega_n &= 4 \text{ rad/s} = .637 \text{ Hz} \\ \zeta &= .2 \end{aligned} \tag{7.6}$$

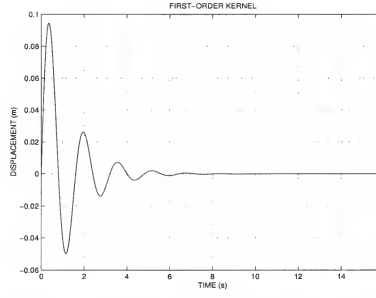


Figure 7.1: Linear Oscillator: First-Order Volterra Kernel

The dynamics of this linear system can be completely described in terms of the first-order Volterra kernel. For a linear system, the first-order kernel is equivalent to the impulse response of the system. The impulse response of equation (7.1) is a well-known result from vibration theory [116] and takes the form

$$h_1(t) = \frac{1}{m\omega_d} e^{-\zeta\omega_n t} \sin(\omega_d t) \tag{7.7}$$

where ω_d is the damped natural frequency, defined as

$$\omega_d = \omega_n \sqrt{1-\zeta^2} \tag{7.8}$$

In this example, $\omega_d = 3.92$ rad/s or .624 Hz. The first-order kernel for this system is depicted in Figure (7.1).

The wavelet-based kernel identification algorithms are used to extract the first-order kernel from simulated input/output data from the system. The input to the system is chosen to be a signal composed of three sinusoids:

$$u(t) = A_1 \sin(2\pi\Omega_1 t) + A_2 \sin(2\pi\Omega_2 t) + A_3 \sin(2\pi\Omega_3 t) \tag{7.9}$$

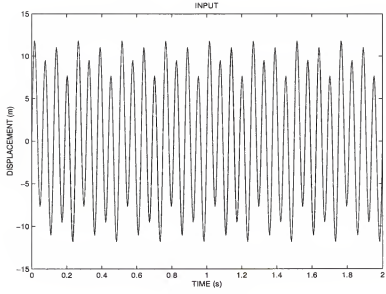


Figure 7.2: Linear Oscillator: Input

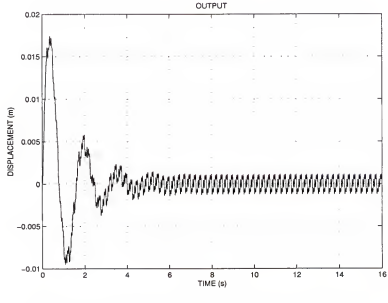


Figure 7.3: Linear Oscillator: Simulated Output

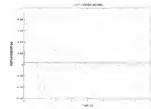
where

$$\begin{aligned} \{A_1, A_2, A_3\} &= \{1, 2, 10\} \text{ N} \\ \{\Omega_1, \Omega_2, \Omega_3\} &= \{4, 8, 16\} \text{ Hz} \end{aligned} \tag{7.10}$$

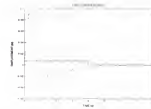
This input is shown in Figure (7.2). For clarity, only the first 2 seconds of the input have been plotted. The corresponding output is obtained via a MATLAB simulation and is depicted in Figure (7.3). In total, there are 16 seconds of input/output data sampled at 256 Hz. Therefore, the discretization level for the zero-order hold approximation of the input and output is $j = 8$. From Figure (7.1), it is apparent that the first-order kernel decays to zero in approximately 8 seconds. Therefore, the kernel length is set to 8 seconds in the kernel identification algorithms. The identified first-order kernels from the Haar identification are shown in Figure (7.4) for a varying number of nonzero terms. Recall that the number of nonzero terms is dependent on the number of resolution levels included in the multiscale approximation of the kernel, as well as the time duration of the identified kernel. Similar plots of the identified kernels from the multiwavelet implementation are shown in Figures (7.5) through (7.7). These figures correspond to the kernel identification using the piecewise-linear, quadratic, and cubic multiwavelets, respectively. For each of the kernel plots, the resolution level of the identified kernel is given as well as the corresponding number of nonzero terms. The coarsest possible level in the multiscale representation of the kernels is level $j_0 = -3$, which corresponds to a grid size of $2^{-3} = 8$ seconds, the duration of the identified kernels. Then, as an example, a level 5 approximation of the kernel corresponds to a single-scale expansion in terms of scaling functions on level 5 or, equivalently, a multilevel expansion in terms of wavelets on levels 4 through -3 and scaling functions on level -3. In each of the plots, the identified kernel is represented by a solid line while the analytical kernel is depicted with a dashed line. The output

predicted by each of the identified kernels is shown in Figures (7.8) through (7.11). For clarity, only the first 8 seconds of identified output is shown. Once again, in each plot the solid line denotes the predicted output while the dashed line represents the actual simulated output.

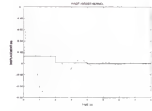
In all cases, the identified kernels and the corresponding outputs converge to the true analytical results as the number of nonzero terms is increased. This is expected since the results should improve as more levels of resolution are included in the multiscale representation of the kernel. This example illustrates the effect of the approximation order of the wavelets on the identified kernels and outputs. From Figure (7.4), it can be seen that the Haar identification of the kernel begins to resemble the form of the analytical kernel when the resolution level is increased to 2, corresponding to 32 nonzero terms in the representation. The identified kernel does not closely match the analytical kernel, however, until the resolution level is increased to 5, a 256-term approximation. This fact is mirrored in the corresponding identified outputs in Figure (7.8) where the identified output does not converge to the actual output until a resolution level of 5 is reached. It is interesting to note that, although the error in the 128-term identification of the kernel does not appear to be large, the resulting identified output clearly shows significant error. This error occurs mainly in the first 2 seconds of the output, which is influenced only by the first 2 seconds of the identified kernel. This is the portion of the identified kernel that is most in error. That, coupled with the relatively high frequency of the input signal, results in the high frequency oscillations in the beginning of the identified output. The multiwavelet identifications of the first-order kernel, shown in Figures (7.5) through (7.7), demonstrate that, compared to the Haar identification, fewer levels of resolution are needed to converge to the analytical kernel. This can be attributed to the fact that the multiwavelets possess higher approximation order than the Haar wavelet. The piecewise-linear multiwavelet identification resembles the analytical kernel fairly well



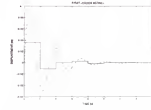
Level -3 Identification (1 Term)



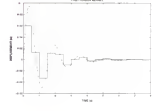
Level -2 Identification (2 Terms)



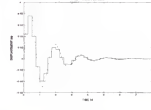
Level -1 Identification (4 Terms)



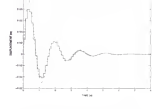
Level 0 Identification (8 Terms)



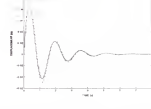
Level 1 Identification (16 Terms)



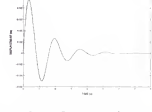
Level 2 Identification (32 Terms)



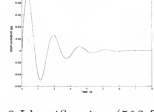
Level 3 Identification (64 Terms)



Level 4 Identification (128 Terms)

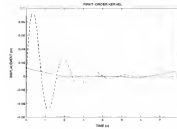


Level 5 Identification (256 Terms)

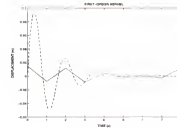


Level 6 Identification (512 Terms)

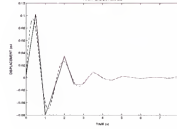
Figure 7.4: Haar Wavelet Identification of the First-Order Volterra Kernel



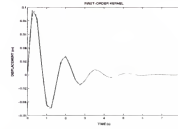
Level -3 Identification (5 Terms)



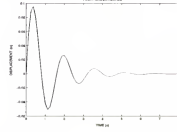
Level -2 Identification (9 Terms)



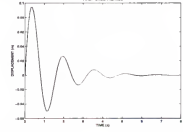
Level -1 Identification (17 Terms)



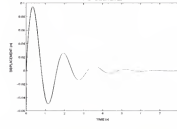
Level 0 Identification (33 Terms)



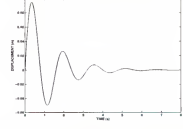
Level 1 Identification (65 Terms)



Level 2 Identification (129 Terms)

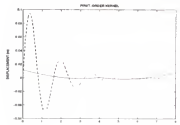


Level 3 Identification (257 Terms)

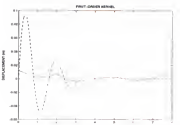


Level 4 Identification (513 Terms)

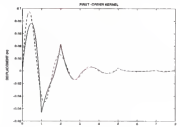
Figure 7.5: Piecewise-Linear Multiwavelet Identification of the First-Order Volterra Kernel



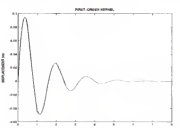
Level -3 Identification (5 Terms)



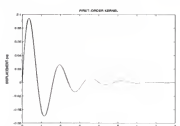
Level -2 Identification (9 Terms)



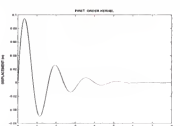
Level -1 Identification (17 Terms)



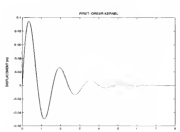
Level 0 Identification (33 Terms)



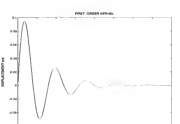
Level 1 Identification (65 Terms)



Level 2 Identification (129 Terms)

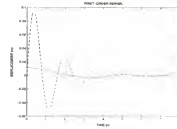


Level 3 Identification (257 Terms)

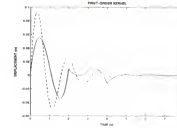


Level 4 Identification (513 Terms)

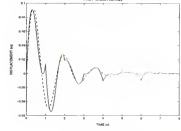
Figure 7.6: Piecewise-Quadratic Multiwavelet Identification of the First-Order Volterra Kernel



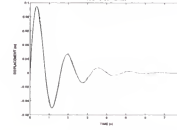
Level -3 Identification (6 Terms)



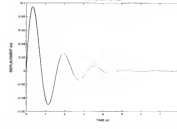
Level -2 Identification (11 Terms)



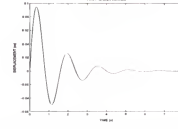
Level -1 Identification (21 Terms)



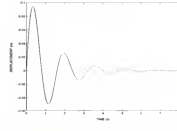
Level 0 Identification (41 Terms)



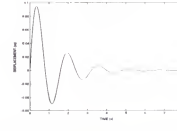
Level 1 Identification (81 Terms)



Level 2 Identification (161 Terms)

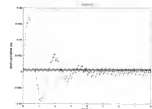


Level 3 Identification (321 Terms)

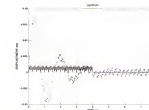


Level 4 Identification (641 Terms)

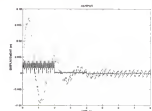
Figure 7.7: Piecewise-Cubic Multiwavelet Identification of the First-Order Volterra Kernel



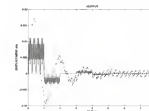
Level -3 Identification (1 Term)



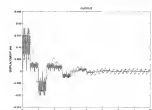
Level -2 Identification (2 Terms)



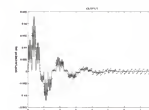
Level -1 Identification (4 Terms)



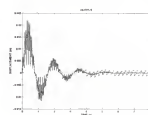
Level 0 Identification (8 Terms)



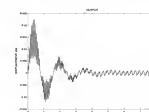
Level 1 Identification (16 Terms)



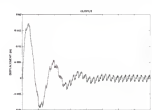
Level 2 Identification (32 Terms)



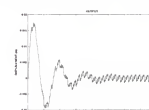
Level 3 Identification (64 Terms)



Level 4 Identification (128 Terms)

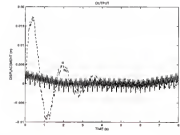


Level 5 Identification (256 Terms)

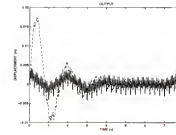


Level 6 Identification (512 Terms)

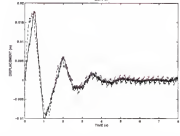
Figure 7.8: Haar Wavelet Identification of the Output



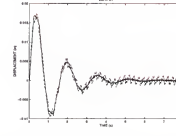
Level -3 Identification (5 Terms)



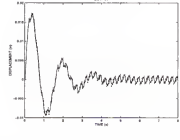
Level -2 Identification (9 Terms)



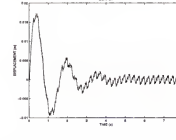
Level -1 Identification (17 Terms)



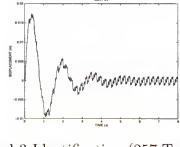
Level 0 Identification (33 Terms)



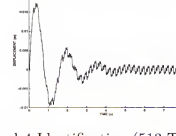
Level 1 Identification (65 Terms)



Level 2 Identification (129 Terms)

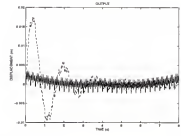


Level 3 Identification (257 Terms)

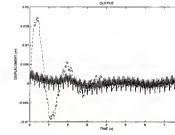


Level 4 Identification (513 Terms)

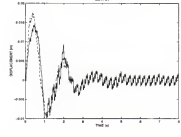
Figure 7.9: Piecewise-Linear Multiwavelet Identification of the Output



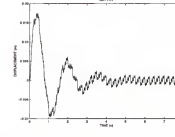
Level -3 Identification (5 Terms)



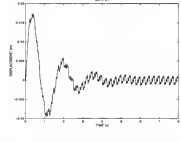
Level -2 Identification (9 Terms)



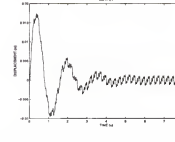
Level -1 Identification (17 Terms)



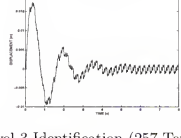
Level 0 Identification (33 Terms)



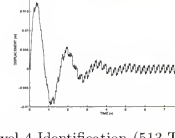
Level 1 Identification (65 Terms)



Level 2 Identification (129 Terms)

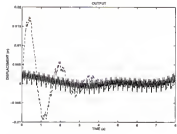


Level 3 Identification (257 Terms)

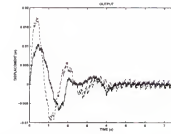


Level 4 Identification (513 Terms)

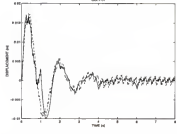
Figure 7.10: Piecewise-Quadratic Multiwavelet Identification of the Output



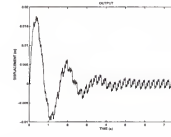
Level -3 Identification (6 Terms)



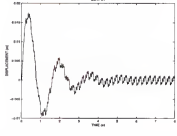
Level -2 Identification (11 Terms)



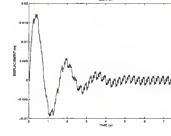
Level -1 Identification (21 Terms)



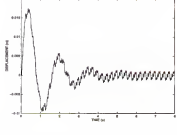
Level 0 Identification (41 Terms)



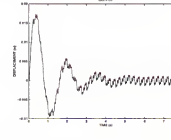
Level 1 Identification (81 Terms)



Level 2 Identification (161 Terms)



Level 3 Identification (321 Terms)



Level 4 Identification (641 Terms)

Figure 7.11: Piecewise-Cubic Multiwavelet Identification of the Output

once the resolution level is increased to 0, corresponding to 33 nonzero terms. Once the resolution level is increased to 2 (129 terms), the identified kernel can no longer be distinguished from the analytical kernel in the figure. The piecewise-quadratic multiwavelet identification shows slightly improved results. The level 0 (33-term) identification of the kernel closely resembles the analytical kernel and the level 2 (65-term) identification cannot be discerned from the analytical kernel in the figure. Similar results are observed for the piecewise-cubic multiwavelet identification of the kernel in Figure (7.7) where, once again, the level 0 (41-term) identification closely matches the analytical kernel and the level 2 (161-term) identification completely converges to it. From the plots of the piecewise-linear identification of the output in Figure (7.9), it can be observed that the level 1 identification matches the actual output fairly well but does not completely converge until the resolution level is increased to 3 (257 terms). On the other hand, the piecewise-quadratic and piecewise-cubic output identifications, shown in Figures (7.10) and (7.11), both closely resemble the actual output at a resolution level of 0. In both cases, once the resolution level is increased to 1, the identified output completely converges to the actual output. In comparing the Haar results to the multiwavelet results, it is clear that fewer levels of resolution are required for kernel identification with the multiwavelets. However, at each resolution level, the multiwavelet identification involves more terms since there are multiple wavelets. In any event, both wavelet-based algorithms provide relatively low-order estimates of the first-order kernel. In comparison, a simple discrete model of an 8-second first-order kernel using input/output data that has been sampled at 256 Hz would require a total of 2048 discrete coefficients.

As a final note, in this example, it is straightforward to select the optimal kernel length since the first-order kernel has a known analytical form. In practice, this information is usually not available. Therefore, it is of interest to consider what happens if a different kernel length is chosen. First, suppose that the kernel length is

selected to be 16 seconds, the full duration of the data set. Figure (7.12) shows the resulting identified kernel in terms of piecewise-quadratic multiwavelets at resolution level 3, corresponding to 257 nonzero terms. Clearly, there is no harm in choosing a kernel length that is longer than necessary other than the fact that computational effort is needlessly spent identifying many zero-valued coefficients. Now, consider the case in which the kernel length is selected to be 4 seconds. A level 3 (33 term) piecewise-quadratic multiwavelet identification of the kernel is shown in Figure (7.13) in comparison to the first 4 seconds of the analytical kernel. The identified kernel matches the analytical kernel well due to the fact that very little is lost when the kernel is truncated at 4 seconds. The resulting identified output, shown in Figure (7.14), has a small amount of error due to the truncation of the kernel. Finally, consider the effect of choosing a kernel length of 2 seconds. In this case, a significant portion of the kernel has been lost. The piecewise-quadratic identification of the kernel at resolution level 3 (17 terms), shown in Figure (7.15), still matches the analytical kernel fairly well, except at the end where it begins to diverge. The identified output, depicted in Figure (7.16), shows significant error, however, due to the truncation of the kernel. This error does not appear until a time of about 2 seconds, corresponding to the time when the truncated part of the kernel starts to have an influence on the output. It is interesting to note that increasing the number of nonzero terms does not improve the kernel estimate in this case, but rather results in a less stable kernel. This can be attributed to the fact that the identification is attempting to compensate for the missing part of the kernel in fitting the output. As a practical point, the mere fact that the identified 2-second kernel has not decayed to zero, as well as the significant error in the identified output, is a clear indication that a longer kernel length is needed.

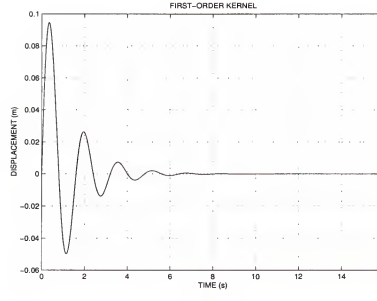


Figure 7.12: Piecewise-Quadratic Identification of the First-Order Kernel (16 sec.)

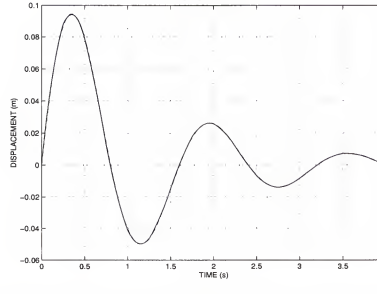


Figure 7.13: Piecewise-Quadratic Identification of the First-Order Kernel (4 sec.)

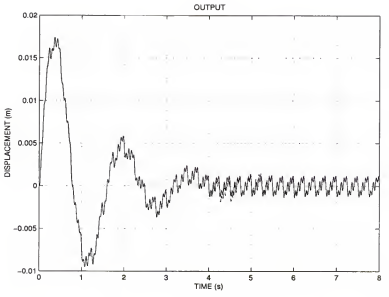


Figure 7.14: Predicted Output from the Identified 4-Second Kernel

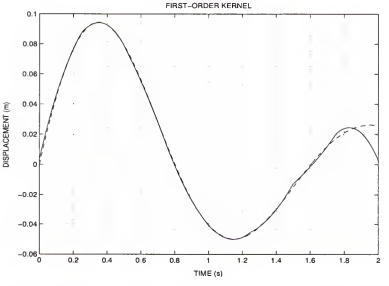


Figure 7.15: Piecewise-Quadratic Identification of the First-Order Kernel (2 sec.)

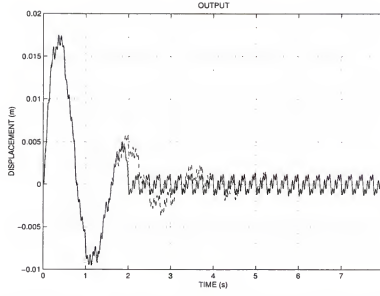


Figure 7.16: Predicted Output from the Identified 2-Second Kernel

7.2 Quadratic Nonlinear System

In this section, a system with a quadratic nonlinearity is considered. Because the system dynamics are completely characterized by the second-order Volterra kernel, it serves as a good validation case for the second-order kernel identification algorithms. In the literature [15], [16], it has been demonstrated that the system shown in Figure (7.17) is a quadratic nonlinear system. This system is composed of a linear block with first-order kernel h_1 followed by a block that squares the output v from the first block. The output y of this system is given as

$$\begin{aligned} y(t) &= v^2(t) = \left[\int_0^t h_1(\xi)u(t-\xi)d\xi\right]^2 \\ &= \int_0^t h_1(\xi)u(t-\xi)d\xi \int_0^t h_1(\eta)u(t-\eta)d\eta \\ &= \int_0^t \int_0^t h_1(\xi)h_1(\eta)u(t-\xi)u(t-\eta)d\xi d\eta \end{aligned} \tag{7.11}$$

where u is the input. Equation (7.11) is of the same form as the output of the second-order Volterra operator

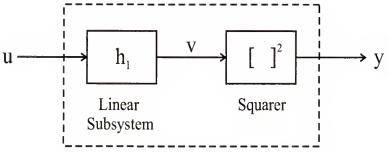


Figure 7.17: Quadratic Nonlinear System

$$y_2(t) = \int_0^t \int_0^t h_2(\xi, \eta) u(t-\xi) u(t-\eta) d\xi d\eta \tag{7.12}$$

where the second-order kernel h_2 is given by

$$h_2(\xi, \eta) = h_1(\xi) h_1(\eta) \tag{7.13}$$

Therefore, it is clear that the system in Figure (7.17) is indeed a quadratic nonlinear system.

In this example, the linear subsystem is taken to have the same form as the linear oscillator that is analyzed in Section 7.1. Recall that the first-order kernel h_1 for such a system is given by

$$h_1(t) = \frac{1}{m\omega_d} e^{-\zeta\omega_n t} \sin(\omega_d t) \tag{7.14}$$

The second-order kernel, from equation (7.13), then takes the form

$$h_2(\xi, \eta) = \left(\frac{1}{m\omega_d}\right)^2 e^{-\zeta\omega_n(\xi+\eta)} \sin(\omega_d \xi) \sin(\omega_d \eta) \tag{7.15}$$

Clearly, this kernel is in symmetric form since $h_2(\eta, \xi) = h_2(\xi, \eta)$. In this example, the parameters for the linear subsystem are taken as

$$m = 2 \text{ kg}$$

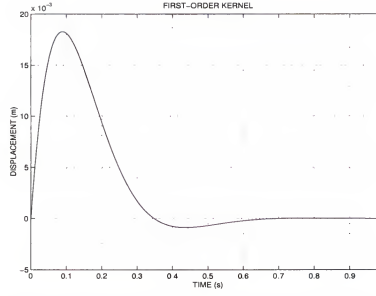


Figure 7.18: Analytical First-Order Kernel for the Linear Subsystem

$$\begin{aligned} c &= 35 \ \text{N} \cdot \text{m/s} \\ k &= 32\pi^2 \ \text{N/m} \end{aligned} \tag{7.16}$$

Then, the corresponding damping ratio, natural frequency, and damped natural frequency have the following values:

$$\begin{aligned} \zeta &= .696 \\ \omega_n &= 12.6 \ \text{rad/s} = 2 \ \text{Hz} \\ \omega_d &= 9.02 \ \text{rad/s} = 1.44 \ \text{Hz} \end{aligned} \tag{7.17}$$

The analytical first-order kernel for the linear subsystem is shown in Figure (7.18). The kernel decays to zero in about 1 second, implying that the size of the second-order kernel can be set to 1 second in the identification. The analytical second-order kernel from equation (7.15) is depicted in Figure (7.19).

The wavelet-based kernel identification algorithms are used to extract the second-order kernel from input/output data from this system. The triangular wavelets are

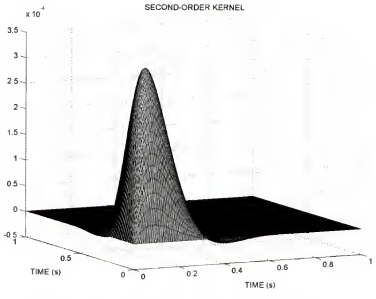


Figure 7.19: Analytical Second-Order Kernel

used to identify the triangular form of the kernel while tensor product multiwavelets are implemented in the identification of the symmetric form of the kernel. The input is chosen to be a linear chirp from 0 to 2 Hz over a 16-second time range. A linear chirp is simply a sinusoidal signal in which the frequency is varied in a linear fashion. The amplitude A of the input force is chosen as 10 N. This input signal and the corresponding output are shown in Figures (7.20) and (7.21) respectively. The input/output data is sampled at 256 Hz, so the resolution level is 8 for the zero-order hold approximation of the input and output. The triangular wavelet identification of the second-order kernel is shown in Figure (7.22) for varying resolution levels. The identified kernels are in triangular form which, as discussed in Section 2.3, can be related to the symmetric kernel $h_{2,sym}$ by the expression [15]

$$h_{2,tri}(\xi,\eta) = 2h_{2,sym}(\xi,\eta)\,\delta_{-1}(\eta-\xi) \tag{7.18}$$

where the unit step function δ_{-1} is defined as

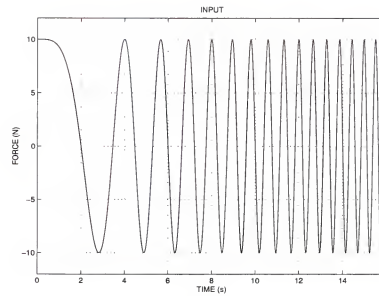


Figure 7.20: Chirp Input Excitation (0-2 Hz)

$$\delta_{-1}(\eta - \xi) := \begin{cases} 1 & \xi \leq \eta \\ 0 & \xi > \eta \end{cases} \tag{7.19}$$

Simply put, the triangular kernel is equal to double the symmetric kernel inside the triangular domain bounded by $0 \leq \xi \leq \eta$, $0 \leq \eta \leq 1$ and zero outside of the domain. Figures (7.23) through (7.25) depict the identified kernels using piecewise-linear, quadratic, and cubic multiwavelets respectively. Also, the analytical kernel is shown for comparison. It should be noted that, because the multiwavelet kernels are symmetric, the number of unique terms that specify the kernels is less than the number indicated in the figures. In general, then, N_2 nonzero terms corresponds to $\frac{1}{2}(N_2^2 + N_2)$ unique terms due to the symmetry. The output corresponding to the identified triangular kernels is plotted in Figure (7.26) while the output predicted by the symmetric kernels, identified in terms of the multiwavelets, is shown in Figures (7.27) through (7.29). For clarity only the first 8 seconds of the output have been plotted. The identified output is depicted by a solid line while the actual simulated output is represented by a dashed line.

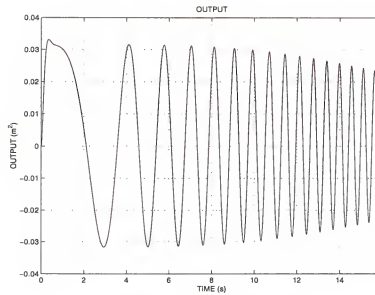
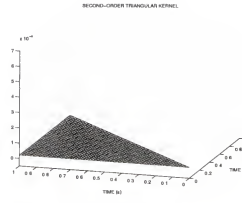
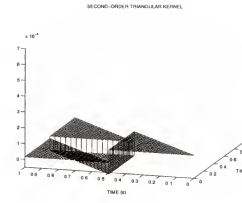


Figure 7.21: Simulated Output

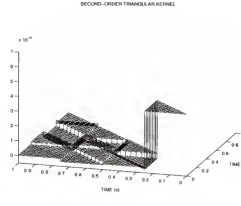
In general, the identified second-order kernels converge to the analytical kernel as the number of resolution levels is increased. It is immediately apparent, however, that the results do not show an exact convergence of the identified kernels. This can be attributed to the fact that the second-order model in both wavelet implementations is more ill-conditioned than the first-order model. Despite regularizing the problem using the truncated singular value decomposition, exact results are difficult to obtain. Nevertheless, the identified kernels clearly have the proper form and converge to the correct amplitude. The triangular wavelet results are somewhat more difficult to evaluate than the multiwavelet results. From Figure (7.22), the triangular kernels appear to converge to a structure that is similar in form to the analytical kernel with twice the amplitude. The finest-resolution kernel is still relatively coarse, but due to computational limits in MATLAB, it is currently not possible to increase the resolution any further. The identified output from the triangular kernels, shown in Figure (7.26), converges well to the actual output. This implies that more resolution is not necessary to gain a useful identification of the triangular kernel. The kernels identified using the piecewise-linear, quadratic, and cubic multiwavelets all exhibit



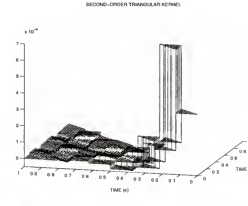
Level 0 Identification (1 Term)



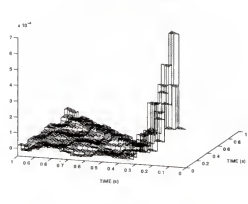
Level 1 Identification (4 Terms)



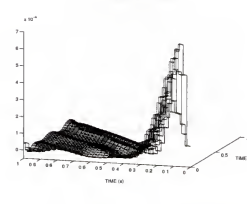
Level 2 Identification (16 Terms)



Level 3 Identification (64 Terms)

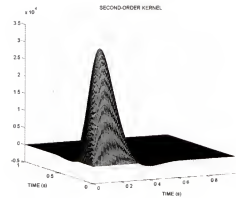


Level 4 Identification (256 Terms)

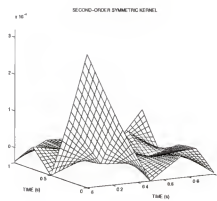


Level 5 Identification (1024 Terms)

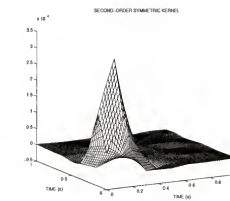
Figure 7.22: Triangular Wavelet Identification of the Second-Order Volterra Kernel



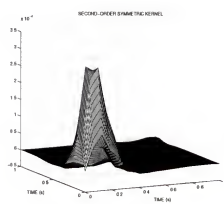
Analytical Second-Order Kernel



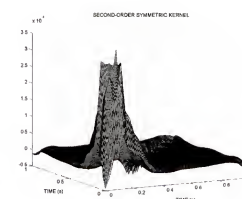
Level 0 Identification (25 Terms)



Level 1 Identification (81 Terms)

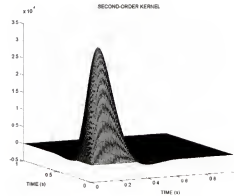


Level 2 Identification (289 Terms)

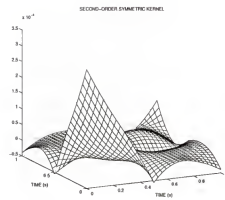


Level 3 Identification (1089 Terms)

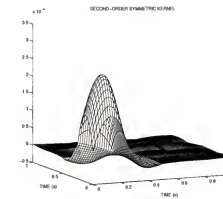
Figure 7.23: Piecewise-Linear Multiwavelet Identification of the Second-Order Volterra Kernel



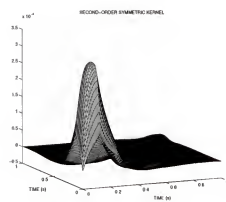
Analytical Second-Order Kernel



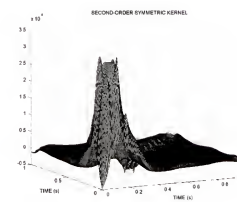
Level 0 Identification (25 Terms)



Level 1 Identification (81 Terms)

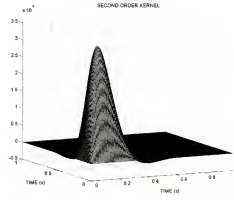


Level 2 Identification (289 Terms)

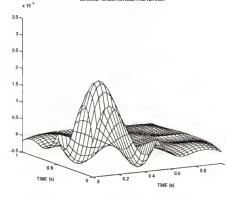


Level 3 Identification (1089 Terms)

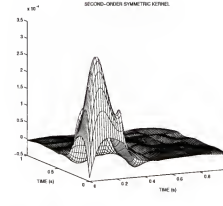
Figure 7.24: Piecewise-Quadratic Multiwavelet Identification of the Second-Order Volterra Kernel



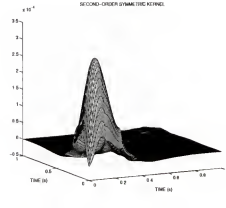
Analytical Second-Order Kernel



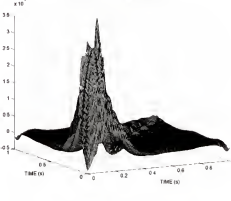
Level 0 Identification (36 Terms)



Level 1 Identification (121 Terms)

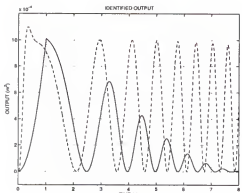


Level 2 Identification (441 Terms)

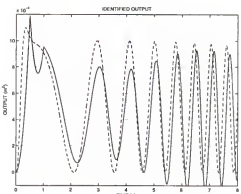


Level 3 Identification (1681 Terms)

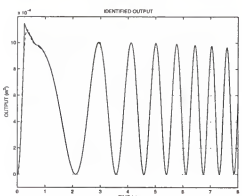
Figure 7.25: Piecewise-Cubic Multiwavelet Identification of the Second-Order Volterra Kernel



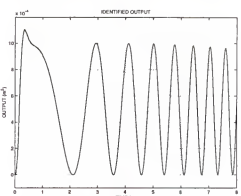
Level 0 Identification (1 Term)



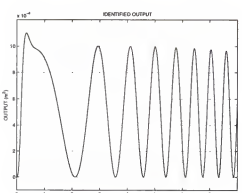
Level 1 Identification (4 Terms)



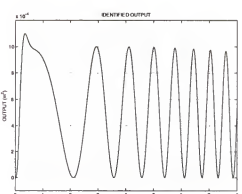
Level 2 Identification (16 Terms)



Level 3 Identification (64 Terms)

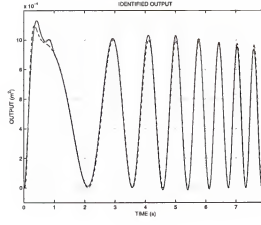


Level 4 Identification (256 Terms)

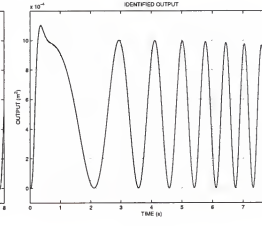


Level 5 Identification (1024 Terms)

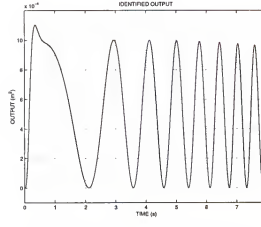
Figure 7.26: Predicted Output from the Identified Triangular Kernels



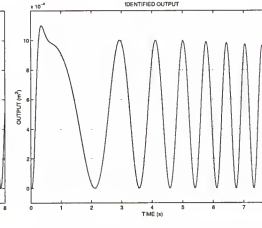
Level 0 Identification (25 Terms)



Level 1 Identification (81 Terms)

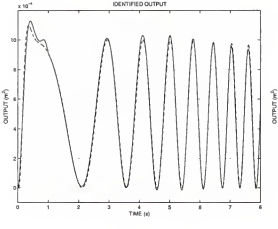


Level 2 Identification (289 Terms)

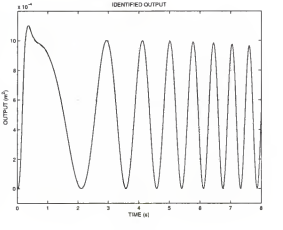


Level 3 Identification (1089 Terms)

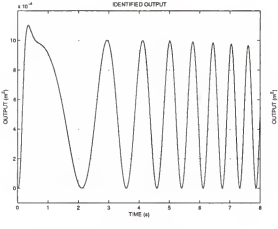
Figure 7.27: Predicted Output from the Piecewise-Linear Multiwavelet Identification of the Second-Order Kernel



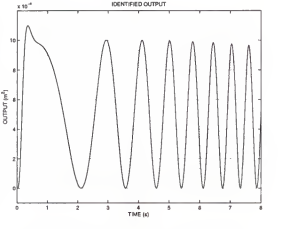
Level 0 Identification (25 Terms)



Level 1 Identification (81 Terms)

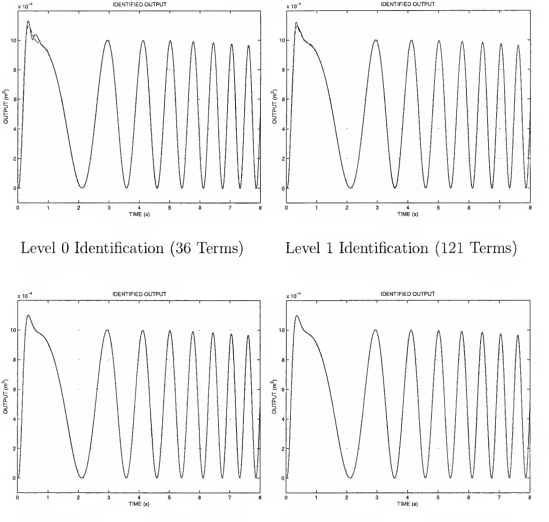


Level 2 Identification (289 Terms)



Level 3 Identification (1089 Terms)

Figure 7.28: Predicted Output from the Piecewise-Quadratic Multiwavelet Identification of the Second-Order Kernel



Level 0 Identification (36 Terms) Level 1 Identification (121 Terms)

Level 2 Identification (441 Terms) Level 3 Identification (1681 Terms)

Figure 7.29: Predicted Output from the Piecewise-Cubic Multiwavelet Identification of the Second-Order Kernel

similar behavior. In all cases, the identified kernels converge to the proper amplitude and form. However, in all three cases, as the resolution level is increased to 3, the identified kernels actually become less accurate than the level 2 kernels. As mentioned before, this is a result of the ill-conditioned nature of the second-order Volterra model. As the number of coefficients in the model is increased, the solution becomes less stable. This is characteristic behavior for an ill-posed inverse problem. Of course, the input has an influence on the model as well. Increasing the number of data points and using a different input could potentially improve the results. At any rate, as shown in Figures (7.27) through (7.29), the identified kernels are capable of reproducing the simulated output of the system. In conclusion, the results show that both wavelet implementations are capable of extracting reasonable estimates of the second-order kernel for this nonlinear system. These estimates are in terms of a relatively small number of coefficients, especially when compared to the fact that a simple discrete model of the same second-order kernel (of 1-second duration) would require $(256)^2$ or $65,536$ terms.

7.3 Nonlinear Oscillator

In previous sections, the kernel identification algorithms have been validated for a linear oscillator and a prototypical quadratic nonlinear system. It has been demonstrated that both wavelet implementations are capable of identifying the analytical Volterra kernels for these simple cases for which it is known that the system dynamics can be completely characterized in terms of the first and second-order kernels, respectively. In this section, a more general case is considered in which both the first and second-order kernels are needed to describe the system dynamics. In addition, the higher-order kernels are nonzero for this nonlinear system. This example lends insight into the ability of the wavelet-based algorithms to extract Volterra kernels for systems with polynomial nonlinearities.

The system analyzed in this example is a nonlinear oscillator which is described by the following equation of motion:

$$m\ddot{y}(t) + c\dot{y}(t) + k_1 y(t) + k_2 y^2(t) = u(t) \tag{7.20}$$

Equation (7.20) takes the same form as the linear oscillator except that a quadratic stiffness term has now been introduced. The system parameters are chosen as

$$\begin{aligned} m &= 1 \text{ kg} \\ c &= 6 \text{ N} \cdot \text{m/s} \\ k_1 &= 4\pi^2 \text{ N/m} \\ k_2 &= 4\pi^2 \text{ N/m} \end{aligned} \tag{7.21}$$

In this case, the linear dynamics of the system are not affected by the nonlinear stiffness term. Therefore, the first-order kernel takes the same form as in the linear oscillator case:

$$h_1(t) = \frac{1}{m\omega_d} e^{-\zeta\omega_n t} \sin(\omega_d t) \tag{7.22}$$

For the values chosen in equation (7.21), the underlying linear system has the following damping ratio, natural frequency, and damped natural frequency:

$$\begin{aligned} \zeta &= .478 \\ \omega_n &= 2\pi \text{ rad/s} = 1 \text{ Hz} \\ \omega_d &= 5.52 \text{ rad/s} = .879 \text{ Hz} \end{aligned} \tag{7.23}$$

The analytical first-order kernel is depicted in Figure (7.30). The nonlinear dynamics are characterized in terms of Volterra kernels of second order and higher. It should be

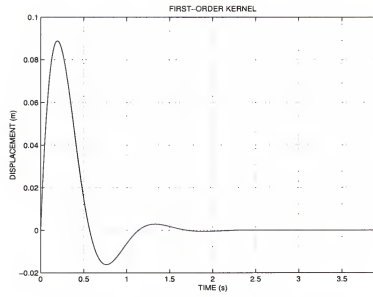


Figure 7.30: Analytical First-Order Kernel

noted that, although the system in equation (7.20) exhibits a quadratic nonlinearity, that does not imply that the nonlinearity can be completely described by the second-order kernel. Indeed, as discussed in Section 2.5, it is possible to derive analytical forms of Volterra kernels from analytic ODEs. It is not difficult to show that the kernels of third order and higher do not vanish for this system. This naturally raises the question of whether or not the second-order kernel is adequate to characterize the nonlinear dynamics. This example investigates under what circumstances an accurate characterization of the nonlinear output is possible while neglecting the effect of the kernels of third order and higher.

The first and second-order kernels are extracted from input/output data from the nonlinear oscillator using the two wavelet-based algorithms. In this example, only the piecewise-quadratic multiwavelets are used in the multiwavelet implementation since the piecewise-linear and cubic multiwavelets tend to give similar results. The system input, shown in Figure (7.31), is chosen to be a linear chirp with a frequency range of 0 to 2 Hz over 32 seconds and an amplitude of 2 N. The response of the system to this input is simulated in MATLAB and plotted in Figure (7.32). The

input/output data is sampled at a rate of 128 Hz, corresponding to a resolution level of 7. First and second-order kernels are extracted from the input/output data using the piecewise-quadratic multiwavelets. These kernels have been truncated after 4 seconds and are depicted in Figures (7.33) and (7.34). In Figure (7.33), the identified first-order kernel at resolution level 4 (257 nonzero terms) is denoted by a solid line and the analytical first-order kernel is represented by a dashed line. The second-order kernel is identified at resolution level 0, corresponding to 289 nonzero terms. The plot of the second-order kernel indicates that it can be further truncated to 2 seconds. Using this information, second-order kernels with 2-second time duration are identified. The extracted kernels at resolution levels 1 (289 terms) and 2 (1089 nonzero terms) are plotted in Figures (7.35) and (7.36) respectively. Similarly, first and second-order kernels are identified using the triangular wavelet algorithm. A Haar wavelet identification of the first-order kernel at resolution level 6 (256 terms) is depicted in Figure (7.37) and an extracted triangular kernel at resolution level 5 (1024 terms) is shown in Figure (7.38). The identified first-order kernels match the analytical kernel relatively well with a small degree of error. The multiwavelet and triangular second-order kernels are in good agreement with each other. The triangular kernel should have double the amplitude of the symmetric kernel over the triangular domain. Figures (7.35) and (7.38) demonstrate that this is indeed the case.

The total output predicted by the identified multiwavelet kernels is plotted in Figure (7.39). Since there is not much difference between the second-order kernel identified at resolution level 1 (289 nonzero terms) in Figure (7.35) and the one identified at resolution level 2 (1089 nonzero terms) in Figure (7.36), the lower-resolution kernel is used to predict the second-order output. The predicted first-order (linear) and second-order (nonlinear) outputs are shown in Figures (7.40) and (7.41) respectively. Since the underlying linear system is known, it is possible to simulate the linear output for comparison with the output predicted by the identified first-order

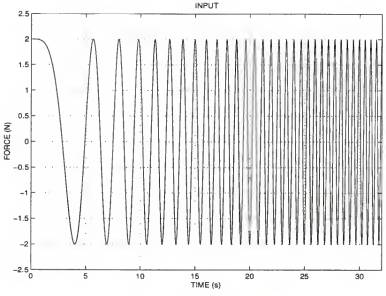


Figure 7.31: Chirp Input (0-2 Hz)

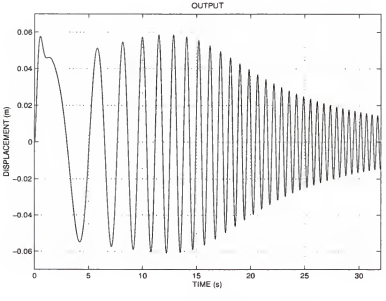


Figure 7.32: Simulated Response to Chirp Input

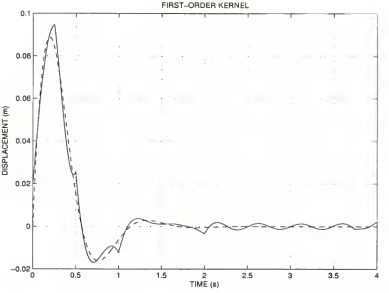


Figure 7.33: Level 4 (257 Term) Identification of the First-Order Kernel Using Piecewise-Quadratic Multiwavelets

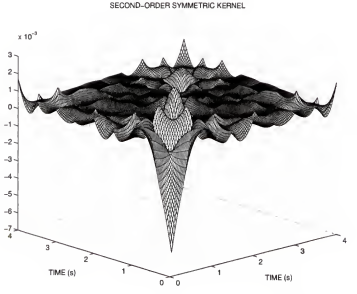


Figure 7.34: Level 0 (289 Term) Identification of the Second-Order Kernel Using Piecewise-Quadratic Multiwavelets

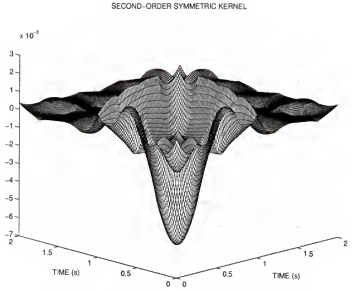


Figure 7.35: Level 1 (289 Term) Identification of the Second-Order Kernel Using Piecewise-Quadratic Multiwavelets

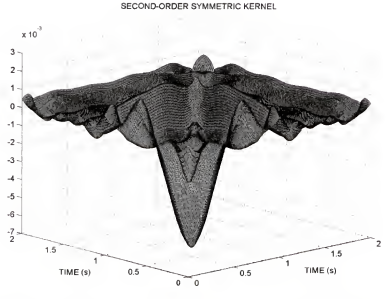


Figure 7.36: Level 2 (1089 Term) Identification of the Second-Order Kernel Using Piecewise-Quadratic Multiwavelets

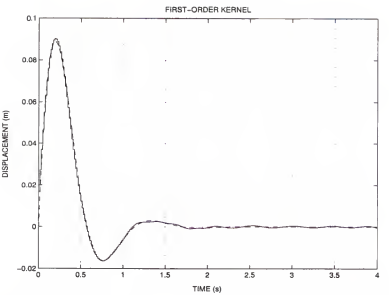


Figure 7.37: Level 6 (256 Term) Identification of the First-Order Kernel Using Haar Wavelets

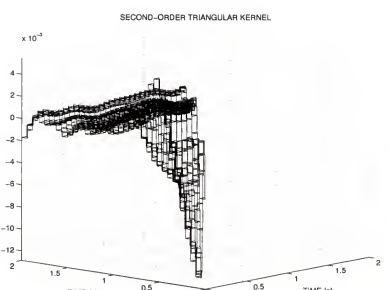


Figure 7.38: Level 5 (1024 Term) Identification of the Second-Order Kernel Using Triangular Wavelets

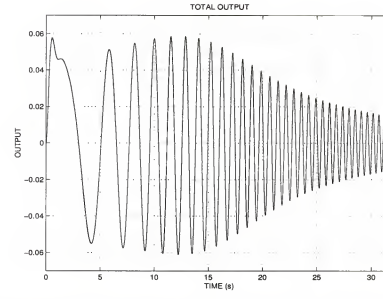


Figure 7.39: Total Output Predicted By the Identified Multiwavelet Kernels

kernel. The nonlinear part of the output is simply the difference between the total simulated output and the simulated linear output. Then, it is possible to evaluate how well the predicted second-order output approximates the nonlinear portion of the simulated output. As can be seen in Figure (7.41), the nonlinear part of the output for this case is at least an order of magnitude smaller than the linear output. It should be noted that the nonlinear dynamics are negative. Therefore, the structure of the identified second-order kernels is physically reasonable since the dominant feature of the kernels is a large negative hump. The total output predicted by the first and second-order kernels extracted using the triangular wavelet implementation is shown in Figure (7.42). The predicted first-order and second-order outputs are displayed in Figures (7.43) and (7.44) respectively. As usual, in all the figures, the solid line denotes the identified output while the dashed line represents the simulated output.

In both wavelet-based kernel identification algorithms, the ability of the extracted kernels to accurately predict the output is not always a guarantee that the kernels are accurate. Because the kernel coefficients are trained to faithfully reproduce the output, there is always the danger that the identified kernels correspond to a mere

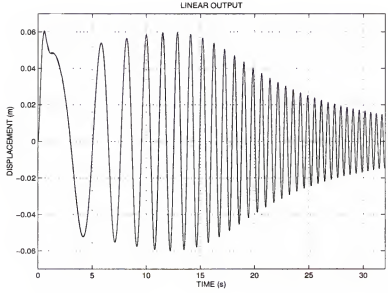


Figure 7.40: First-Order (Linear) Output Predicted By the Identified First-Order Multiwavelet Kernel

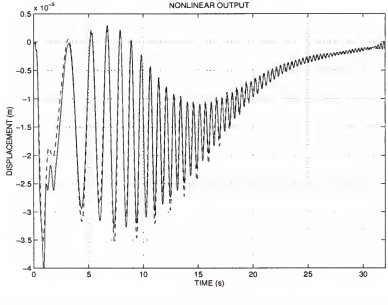


Figure 7.41: Second-Order (Nonlinear) Output Predicted By the Identified Second-Order Multiwavelet Kernel

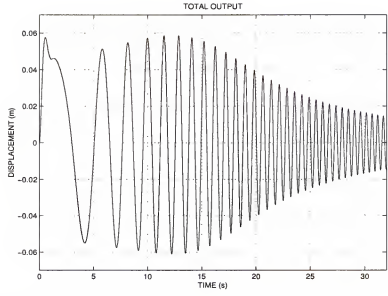


Figure 7.42: Total Output Predicted By the Identified Haar and Triangular Wavelet Kernels

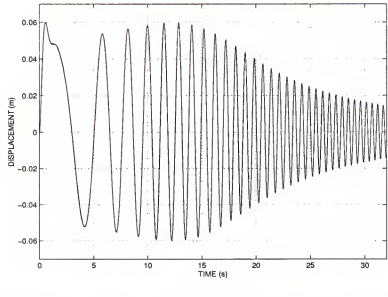


Figure 7.43: First-Order (Linear) Output Predicted By the Identified First-Order Haar Wavelet Kernel

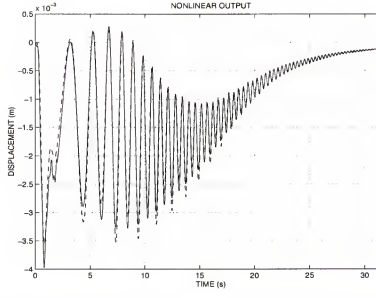


Figure 7.44: Second-Order (Nonlinear) Output Predicted By the Identified Second-Order Triangular Kernel

a curve-fit of the specific data set from which they are extracted. In the examples considered thus far, analytical kernels have been available to serve as a check on the accuracy of the identified kernels. In practice, however, one does not usually have analytical forms of the Volterra kernels. Instead, the kernels can be validated by evaluating their ability to predict the outputs corresponding to different input data sets. In this example, although the analytical first-order kernel is known, and it is possible to derive the analytical second-order kernel from the ODE in equation (7.20), the identified kernels are tested on a number of different input/output data sets. This analysis is performed for the piecewise-quadratic multiwavelet kernels depicted in Figures (7.33) and (7.35). First, an input composed of two sinusoids with frequencies of .5 Hz and 4 Hz is considered. This input, depicted in Figure (7.45), takes the form

$$u_1(t) = A_1 \sin(\pi t) + A_2 \sin(8\pi t) \tag{7.24}$$

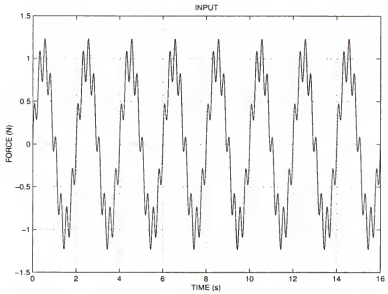


Figure 7.45: Input u_1 for Kernel Validation

where the amplitudes are selected as $A_1 = 1$ N and $A_2 = .25$ N. Although the total duration of the input data set is 32 seconds, only the first 16 seconds are plotted for clarity. The predicted first-order and second-order outputs from the identified kernels are compared with the linear and nonlinear portions of the simulated output in Figures (7.46) and (7.47). Once again, only the first 16 seconds of output are shown. Since the nonlinear part of the output is very small compared to the linear part, the total output differs little from the linear output and is not plotted in this case. The results show that the first-order kernel accurately predicts the linear dynamics. This is not surprising since the identified kernel matches the analytical kernel relatively well. The identified second-order kernel is able to predict the nonlinear dynamics with relatively small error.

Next, consider an input of the form

$$\begin{aligned} u_2(t) &= A_1 \sin\left(\frac{\pi}{16}t\right) + A_2 \sin(\pi t) + A_3 \sin(2\pi t) \\ &\quad + A_4 \sin(4\pi t) + A_5 \sin(16\pi t) \end{aligned} \tag{7.25}$$

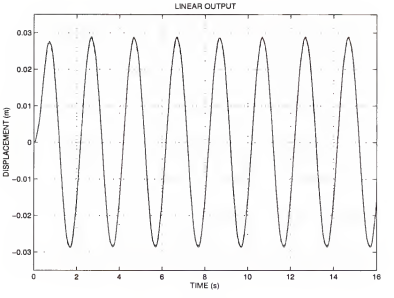


Figure 7.46: Predicted First-Order (Linear) Output for Input u_1

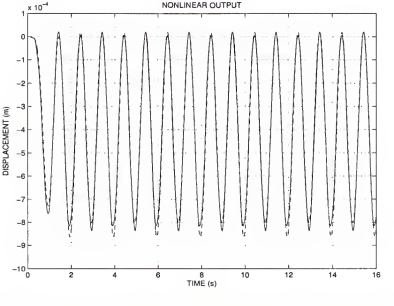


Figure 7.47: Predicted Second-Order (Nonlinear) Output for Input u_1

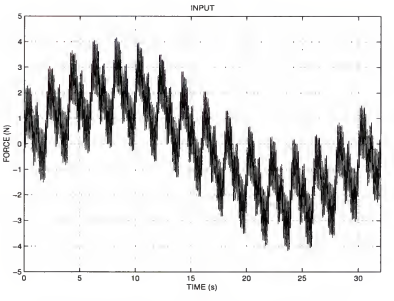


Figure 7.48: Input u_2 for Kernel Validation

which is composed of sinusoids with frequencies of $\{1/32, .5, 1, 2, 8\}$ Hz. The amplitudes $\{A_1, A_2, A_3, A_4, A_5\}$ are chosen as $\{2, 1, .5, .25, 1\}$ N. This input is displayed in Figure (7.48). The predicted first and second-order outputs are shown in Figures (7.49) and (7.50). Once again, the identified first-order kernel predicts the linear dynamics with a high degree of accuracy. The identified second-order kernel captures the nonlinear dynamics relatively well. The difference between the predicted and actual nonlinear dynamics can be attributed in part to error in the identified kernel and, more significantly, the truncation error resulting from neglecting higher-order kernels. The total output predicted by the identified kernels is compared to the simulated output in Figure (7.51).

Next, the amplitude A_1 in the input signal u_1 is changed from 1 to 5. The resulting input u_3 is depicted in Figure (7.52). The increased amplitude of the input has the effect of increasing the influence of the nonlinearity on the overall output. The predicted first and second-order outputs are shown in Figures (7.53) and (7.54) respectively. The nonlinear part of the output is now a factor of 10 smaller than the linear part. In contrast, for the case where $A_1 = 1$, that ratio is greater than 30.

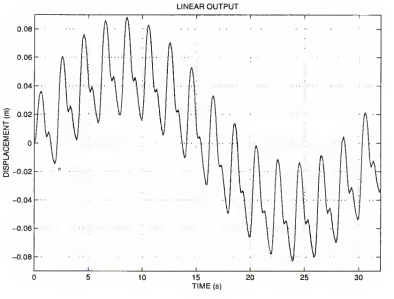


Figure 7.49: Predicted First-Order (Linear) Output for Input u_2

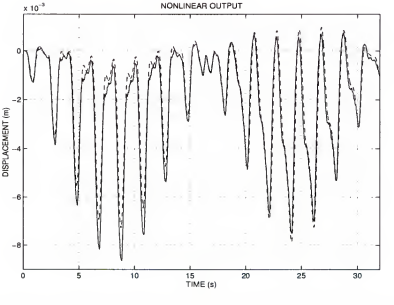


Figure 7.50: Predicted Second-Order (Nonlinear) Output for Input u_2

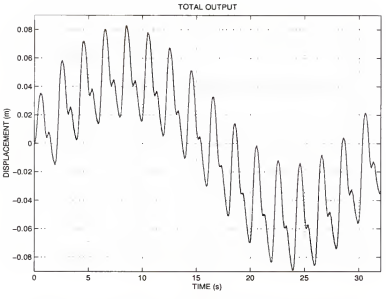


Figure 7.51: Total Predicted Output for Input u_2

Thus, in this case, the nonlinearity exerts a greater influence than before. From Figure (7.53), the first-order kernel accurately predicts the linear dynamics. The predicted second-order output, however, shows significant amplitude error in capturing the nonlinear oscillations. This is due to the fact that the increase in the amplitude of the input has excited the higher-order nonlinear dynamics that have been neglected in the model. Nevertheless, the total output predicted by the identified kernels is still very accurate as can be seen in Figure (7.55).

As a final example, the effect of increasing the input amplitude is investigated further by changing the amplitude A_1 in the input signal u_2 from 2 to 10. The resulting input u_4 is shown in Figure (7.56). The predicted first and second-order outputs are plotted in Figures (7.57) and (7.58) respectively. The nonlinear dynamics are now of the same order of magnitude as the linear dynamics. As can be seen in Figure (7.58), the identified second-order kernel is not capable of accurately predicting the nonlinear dynamics. This can be attributed to the fact that as the input amplitude is increased, the higher-order Volterra operators have an increased influence on the nonlinear output. The truncation error due to neglecting the higher-order kernels is

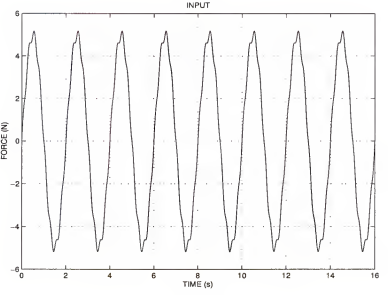


Figure 7.52: Input u_3 for Kernel Validation

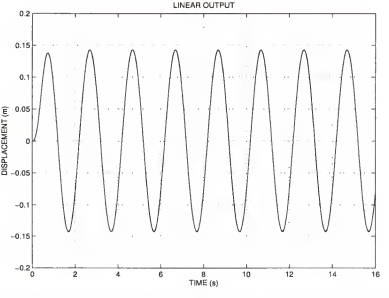


Figure 7.53: Predicted First-Order (Linear) Output for Input u_3

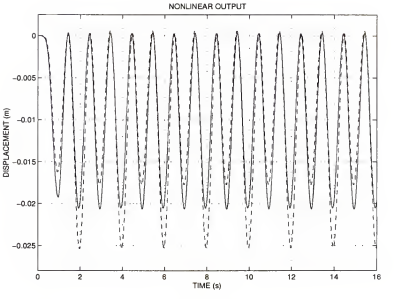


Figure 7.54: Predicted Second-Order (Nonlinear) Output for Input u_3

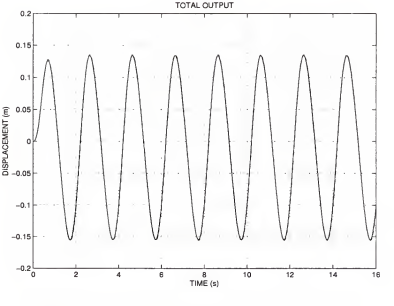


Figure 7.55: Total Predicted Output for Input u_3

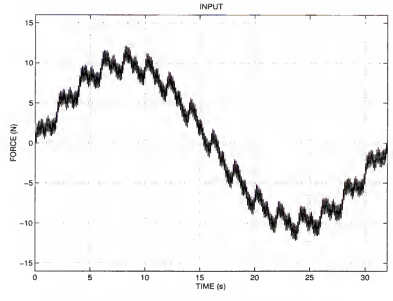


Figure 7.56: Input u_4 for Kernel Validation

magnified and the second-order kernel alone is no longer capable of characterizing the nonlinear dynamics. These validation cases show that the nonlinear dynamics of the system are well characterized by the identified second-order kernel for a range of input amplitudes. As the input amplitude grows larger, however, higher-order kernels have a greater influence and can no longer be neglected. These results are consistent with the convergence discussion given in Section 2.5, where it was noted that the convergence of a truncated Volterra series is guaranteed only for a bounded range of input amplitudes.

7.4 Flutter Analysis

As a final example, Volterra kernels are identified from flight test data from the Aerostructures Test Wing (ATW), which was developed at NASA Dryden to study flutter. Flutter is a linear aeroelastic instability that results from the unfavorable interaction of aerodynamic forces with the wing structure. In classical binary flutter, two modes of vibration, often the bending and torsion modes, become coupled as the air speed approaches the flutter speed. This coupling causes the damping ratio of one of the modes to increase while the damping ratio corresponding to the other

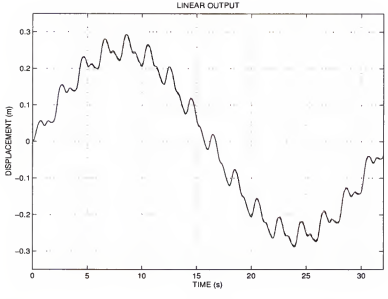


Figure 7.57: Predicted First-Order (Linear) Output for Input u_4

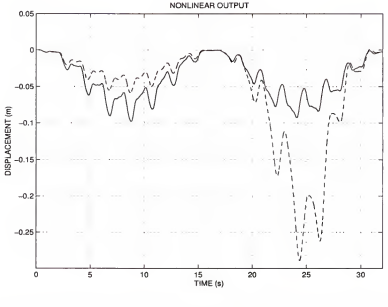


Figure 7.58: Predicted Second-Order (Nonlinear) Output for Input u_4

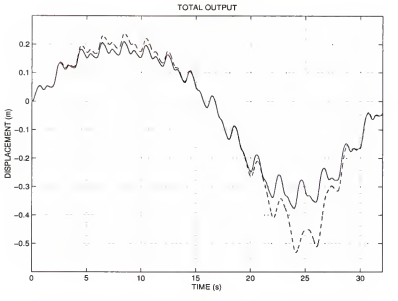


Figure 7.59: Total Predicted Output for Input u_4

mode approaches zero. At the critical flutter speed, this damping ratio reaches zero and any aerodynamic disturbance results in an unstable oscillation in the undamped mode. Thus, flutter is usually catastrophic, and the determination of flutter speeds is a critical aspect of the design of any aircraft. One common approach to flutter prediction is to extrapolate the damping ratios from flight data at different flight conditions to determine the critical flutter speed [117]. Another approach is to employ the concept of a flutter margin to predict the flutter speed [118]. These methods have their limitations, however, and do not always lead to conservative flutter estimates [119]. Recently, Lind and Brenner [6] developed the flutterometer, a flutter prediction tool that computes a flutter speed based on an analytical model. The computed flutter speed is robust with respect to an uncertainty description that is generated from flight data. In this manner, a worst-case flutter speed is predicted that is conservative with respect to the true flutter speed.

The ATW was designed at the NASA Dryden Flight Research Center in order to test the flutterometer. The ATW, pictured in Figure (7.60), consisted of a wing and boom assembly that was attached to the undercarriage of an F-15 aircraft as

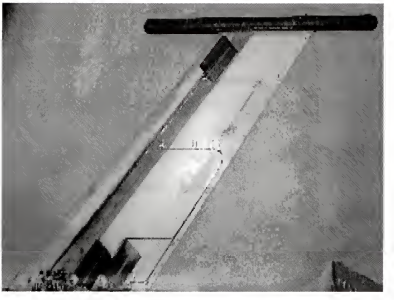


Figure 7.60: Aerostructures Test Wing

shown in Figure (7.61). The wing span of the ATW was 18 in. and the total weight was 2.66 lbs. The ATW was designed to flutter at a speed that was within the safe flight envelope of the host F-15. By actually driving the ATW to flutter, the true flutter speed of the ATW was experimentally determined for comparison with the flutterometer predictions. The wing was constructed from lightweight materials in order to guarantee that it would not damage the F-15 once it broke apart during flutter. The ATW was instrumented with 18 strain gages and 3 accelerometers. Piezoelectric patches were mounted on the upper and lower surfaces of the wing. Acting in tandem, the patches acted as a single actuater that provided a linear chirp excitation from 5 to 35 Hz. Flight tests of the ATW were performed at NASA Dryden in the spring of 2001. The interested reader can refer to Ref. [119] for more details regarding the flight-testing of the ATW.

Flight data from the ATW was taken at a number of different flight conditions. In this example, Volterra kernels are extracted from data corresponding to a flight condition of Mach number .8 and altitude $20,000$ ft. The input is taken as the chirp excitation over a frequency range of 5 to 35 Hz. The output is taken as the

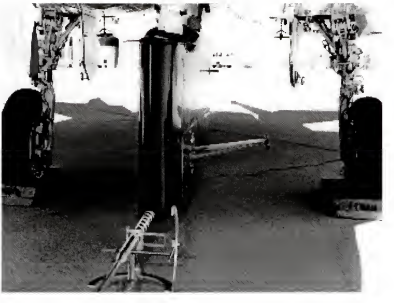


Figure 7.61: Mounting of the Aerostructures Test Wing

response measured by an accelerometer that was mounted on the trailing edge of the boom. The measured output is shown in Figure (7.62). The dominant modes of the system are an 18 Hz bending mode and a 24 Hz torsion mode. The response of these modes can be visualized in Figure (7.62) in that an increased output response is evident at approximately 25 seconds and 37 seconds. These times correspond to when the input excitation frequency reaches 18 and 24 Hz, respectively. The data is sampled at a rate of 800 Hz. A zero-order hold approximation of the input and output at level 8, corresponding to a sampling frequency of 256 Hz, is generated by projecting the data onto Haar scaling functions on level 8. First and second-order Volterra kernels are then identified from the input/output data. Varying the kernel length in the identification, it has been determined that a time duration of 1 second is sufficient for the first-order kernel. A Haar identification of the first-order kernel at a resolution level of 8 (256 nonzero terms) is shown in Figure (7.63). Similarly, a piecewise-quadratic multiwavelet identification of the kernel at resolution level 6 (257 nonzero terms) is shown in Figure (7.64). The two identified kernels are very similar in form. Viewed in the frequency domain, both extracted kernels

show the dominant modes at 18 Hz and 24 Hz, as shown in the frequency domain plots of the kernels in Figures (7.65) and (7.66). Unfortunately, attempts to extract a physically meaningful second-order kernel have proven unsuccessful in both wavelet-based algorithms. The identified second-order kernels show no coherent structure and fail to converge as the number of nonzero wavelet coefficients are increased. Thus, it is suspected that the second-order kernels are modeling noise in the output. The fact that the identified second-order kernels are much smaller in magnitude than the first-order kernels, coupled with the fact that the identified first-order kernels resemble impulse response functions, implies that the ATW was essentially a linear system. This is in agreement with a previous analysis of the ATW that demonstrated that it responded linearly to a range of inputs [120]. Because the two identified first-order kernels are so similar, the predicted linear responses are essentially the same and it is not necessary to show both plots. The linear response predicted by the Haar identification of the first-order kernel is shown in Figure (7.67). Qualitatively, the predicted output appears to capture the response of the two dominant modes at 25 seconds and 37 seconds. The predicted linear output does not identify much of the noise that is present in the original flight data, particularly at the beginning of the data set where the response is almost entirely due to noise.

A practical application of the identified first-order kernels is described in Ref. [33]. In this paper, the kernels are incorporated into the flutterometer in order to estimate more accurate flutter speeds for the ATW. The original flutterometer implementation computes flutter speeds that are robust with respect to an uncertainty description that is generated from flight data. Basically, a transfer function is computed from the nominal finite element model. An uncertainty envelope is then placed around the nominal transfer function in such a manner that the transfer function derived from the flight data is covered. This is demonstrated in Figure (7.68). The uncertainty envelope ensures a conservative flutter estimate, but the computed flutter

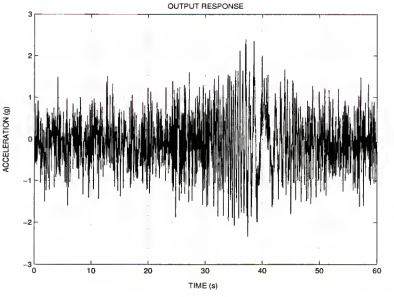


Figure 7.62: Accelerometer Response to a Chirp Excitation

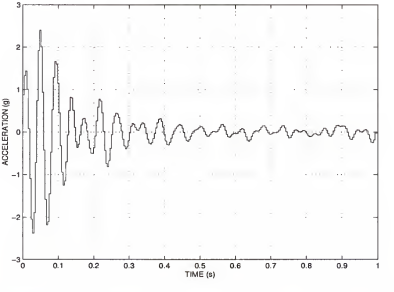


Figure 7.63: Haar Identification of the First-Order Kernel at Resolution Level 8 (256 terms)

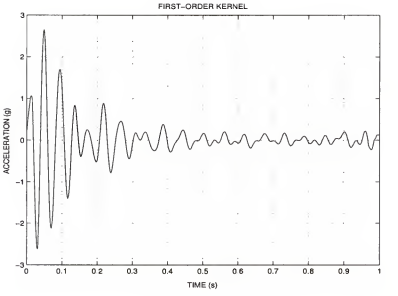


Figure 7.64: Piecewise-Quadratic Multiwavelet Identification of the First-Order Kernel at Resolution Level 6 (257 terms)

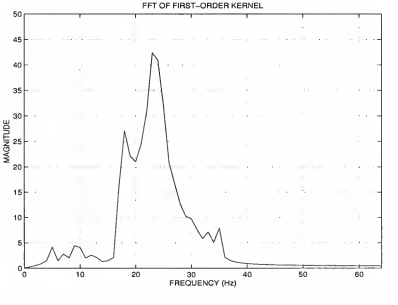


Figure 7.65: FFT of the Haar Identification of the First-Order Kernel

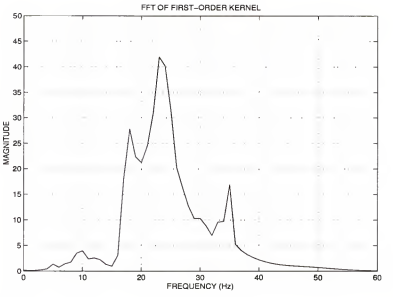


Figure 7.66: FFT of the Piecewise-Quadratic Multiwavelet Identification of the First-Order Kernel

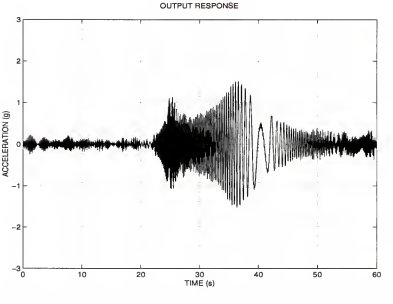


Figure 7.67: Linear Response Predicted by the Identified First-Order Kernel

speed is overly conservative. One of the reasons for this is the fact that in the current flutterometer implementation, the flight data is used to generate the uncertainty description. Flutter is a linear phenomenon and is modeled by an analytical linear model. The flight data, however, consists of linear dynamics, nonlinear dynamics, and noise. Therefore, an uncertainty description based on the differences between the flight data and the nominal model incorporates nonlinear dynamics and noise, which do not contribute to the linear flutter dynamics. Using the first-order kernel that has been identified from the flight data, it is possible to extract the linear dynamics. Then, an uncertainty description can be obtained in terms of the differences between the analytical model and the identified linear dynamics. This leads to a more accurate characterization of the uncertainty in the linear flutter model. This is demonstrated in Figure (7.69), in which the transfer function of the linear dynamics predicted from the Haar identification of the first-order kernel is used to define the uncertainty envelope. Comparing Figures (7.68) and (7.69), the uncertainty envelope based on the extracted linear dynamics is smaller than that obtained from the flight data. This implies that a less conservative flutter speed will be predicted by the flutterometer using the new implementation. This is indeed the case as the flutterometer predicts flutter speeds of 403 knots of equivalent air speed (KEAS) for the uncertainty description based on the flight data and 413 KEAS for the uncertainty description obtained from the first-order Volterra kernel. By design, the flutterometer always estimates a more conservative flutter speed than the nominal model, which in this case predicts a flutter speed of 431 KEAS. The true flutter speed of the ATW, determined in flight test when the ATW was driven to flutter and destroyed, is 460 KEAS. Thus, the first-order Volterra kernel identification can be used to improve the accuracy of the flutterometer and, in many cases, reduce some of the inherent conservatism in the resulting flutter estimates.

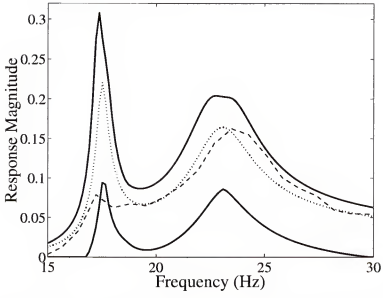


Figure 7.68: Uncertainty Description Based on the Flight Data: flight data (– – –), nominal model (···), uncertainty envelope (—)

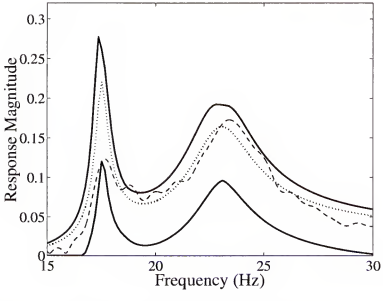


Figure 7.69: Uncertainty Description Based on the First-Order Volterra Kernel: identified linear dynamics (– – –), nominal model (···), uncertainty envelope (—)

CHAPTER 8
CONCLUSION

In this dissertation, two novel wavelet-based algorithms have been developed for the identification of Volterra kernels for nonlinear dynamical systems. In both cases, the Volterra series model is truncated to include only the first and second-order Volterra operators. The first implementation employs Haar wavelets for the representation of the first-order kernel and triangular wavelets for the approximation of the triangular form of the second-order kernel. These triangular wavelets have been constructed using the method developed by Micchelli and Xu [4, 5] for deriving wavelets over finite sets. The second wavelet implementation uses piecewise-polynomial multiwavelets to approximate the first and second-order kernels. These multiwavelets have been generated from classical finite element basis functions using the technique of successive intertwinings of multiresolution analyses, developed by Donovan, Geronimo, and Hardin [2, 3]. This method has been demonstrated for the construction of piecewise-quadratic and piecewise-cubic multiwavelets. In addition, it has been shown that this approach can be generalized to construct multiwavelets of arbitrary order from the appropriate finite elements, and a theorem has been given to that effect. These piecewise-polynomial multiwavelets are easily adapted to the finite domains over which the kernels are supported. In this case, tensor product multiwavelets are used to represent the symmetric form of the second-order kernel.

These wavelet-based algorithms have been implemented to extract Volterra kernels from input/output data from a number of dynamical systems. The results show that both wavelet approaches were able to successfully identify first-order kernels for the linear oscillator case and second-order kernels for the prototypical quadratic

nonlinear system in Section 7.2. In these examples, relatively low-order estimates of the kernels were obtained. The identified first-order kernels in the linear oscillator case converged exactly to the analytical kernel. In the quadratic example, however, the identified kernels converged but showed some instability as the number of coefficients was increased in the model. This is most likely due to the ill-posed nature of the second-order identification. This was reflected in the singular value plots, which typically showed a large number of singular values that were essentially zero.

The nonlinear oscillator example demonstrated the ability of the wavelet algorithms to simultaneously identify first and second-order kernels for a system with a polynomial nonlinearity. The extracted kernels were not completely smooth, but clearly converged to a specific form. The identified first-order kernels matched the analytical first-order kernel from the underlying linear system relatively well. The identified multiwavelet and triangular second-order kernels were in agreement, lending confidence in the results. When identifying Volterra kernels from input/output data, there is always a danger that the identified kernels correspond to a mere curve-fit of the data rather than the true structure of the system. Therefore, the identified kernels were validated by testing their ability to predict the outputs corresponding to different inputs. The kernels were able to successfully predict these outputs for a range of input amplitudes. However, as the input amplitude was increased, the identified second-order kernel was no longer sufficient to characterize the nonlinear dynamics. This is due to the fact that increasing the input amplitude resulted in greater contributions from the unmodeled, higher-order kernels. Therefore, it is important to note that, in many cases, a model based on a truncated Volterra series may be valid only for a limited range of input amplitudes.

Finally, Volterra kernels were extracted from flight test data from the Aerostructures Test Wing (ATW). It was demonstrated that first-order kernels can be identified from noisy flight data. The identified kernels successfully captured the two dominant

modes of the system. Attempts to identify a convergent second-order kernel were unsuccessful, however. This could be explained by the fact that the ATW is basically a linear system. Therefore, the second-order kernel can be expected to be very small and the high level of noise in the data may have prevented the algorithms from extracting a kernel. An application of the identified first-order kernels to flutter prediction has also been given.

The wavelet-based algorithms have several limitations. First, the current implementations in MATLAB have a limit as to the number of coefficients that can be included in the model. This imposes a limit on the resolution of the identified kernels. In most of the examples, this was not a problem as it was demonstrated that the kernels could be represented in terms of a relatively small number of coefficients. In some cases, such as the quadratic nonlinear system discussed in Section 7.2, increasing the number of coefficients can actually make it more difficult to obtain a stable solution. The limit on the number of coefficients is a serious liability, however, if the system has a long memory. In that case, the finest possible resolution level may be very coarse since the corresponding kernels have longer duration. Therefore, in the current implementations, there are practical limits on the lengths of the identified kernels. In general, for a sampling rate of 128 Hz for the input and output, the algorithms have difficulty dealing with kernel lengths that exceed about 32 seconds in the first-order case and approximately 8 seconds in the second-order case. Both algorithms require a significant amount of computational time. This is especially true for the triangular wavelet implementation. In all cases, the computational time increases with the length of the identified kernels. The multiwavelet implementation is much faster than the triangular implementation. For the examples in this dissertation, in which the kernels were of relatively short time duration, the second-order part of the multiwavelet algorithm took on the order of 5 to 10 minutes to run while the triangular algorithm required on the order of an hour. Another limitation of

the wavelet algorithms is that it is extremely difficult to extract the kernels in some cases. In the nonlinear oscillator case, the identified second-order kernels were extremely sensitive to the point where the singular values were truncated in generating the solution. Therefore, extracting the kernels was a tedious procedure, requiring a significant amount of trial and error in truncating the singular values. In the ATW case, convergent second-order kernels were not found, presumably due to the fact that the second-order kernel was very small for that system and noise precluded the identification of the kernel.

Despite its limitations, the wavelet-based kernel identification algorithms have shown considerable promise for the identification of first and second-order kernels for weakly nonlinear systems. There are many possible directions for future work. One such area involves improving the current implementations to make them more efficient and to increase the number of coefficients that can be included in the model. An important step that must be taken is that second-order kernels need to be identified from experimental data. This is essential as the goal of this work is to provide a nonlinear modeling tool that can be used in engineering practice. Another possibility for future work is to extend these algorithms to include higher-order kernels. This would clearly require an increase in the computational capabilities of the current implementations. There is also a great deal of work that can be done in terms of the practical application of these algorithms. As discussed in Section 7.4, identified first-order kernels from flight data can be incorporated into the flutterometer to achieve more accurate flutter predictions. A natural extension of this application, then, would be the identification of second-order kernels for aeroelastic systems in order to study and predict nonlinear phenomena such as limit cycle oscillation.

REFERENCES

[1] G. Beylkin, R. Coifman, and V. Rokhlin, "Fast wavelet transforms and numerical algorithms," *Comm. Pure and Applied Mathematics*, vol. 44, pp. 141–183, 1991.

[2] G. Donovan, J. Geronimo, and D. Hardin, "Intertwining multiresolution analyses and the construction of piecewise-polynomial wavelets," *SIAM Journal of Mathematical Analysis*, vol. 27, no. 6, pp. 1791–1815, Nov. 1996.

[3] G. Donovan, J. Geronimo, and D. Hardin, "Orthogonal polynomials and the construction of piecewise-polynomial smooth wavelets," *SIAM Journal of Mathematical Analysis*, vol. 30, no. 5, pp. 1029–1056, 1999.

[4] C. Micchelli and Y. Xu, "Using the matrix refinement equation for the construction of wavelets on invariant sets," *Applied and Computational Harmonic Analysis*, vol. 1, pp. 391–401, 1994.

[5] C. Micchelli and Y. Xu, "Reconstruction and decomposition algorithms for biorthogonal multiwavelets," *Multidimensional Systems and Signal Processing*, vol. 8, pp. 31–69, 1997.

[6] R. Lind and M. Brenner, "The flutterometer: An on-line tool to predict robust flutter margins," *Journal of Aircraft*, vol. 37, no. 6, pp. 1105–1112, Nov.-Dec. 2000.

[7] V. Volterra, *Theory of Functionals and of Integral and Integro-Differential Equations*, Dover Publications, Inc., New York, 1959.

[8] S. Boyd and L.O. Chua, "Fading memory and the problem of approximating nonlinear operators with Volterra series," *IEEE Trans. on Circuits and Systems*, vol. CAS-32, no. 11, pp. 1150–1161, November 1985.

[9] Y.H. Ku and A.A. Wolf, "Volterra-Wiener functionals for the analysis of nonlinear systems," *Journal of the Franklin Institute*, vol. 281, no. 1, pp. 9–26, January 1966.

[10] S. Boyd, L.O. Chua, and C.A. Desoer, "Analytical foundations of Volterra series," *IMA Journal of Mathematical Control & Information*, vol. 1, pp. 243–282, 1984.

[11] I. Sandberg, "Uniform approximation with doubly finite Volterra series," *IEEE Trans. on Signal Processing*, vol. 40, no. 6, pp. 1438–1441, June 1992.

[12] M. Fliess, M. Lamnabhi, and F. Lamnabhi-Lagarrigue, "An algebraic approach to nonlinear functional expansions," *IEEE Trans. on Circuits and Systems*, vol. CAS-30, no. 8, pp. 554–570, August 1983.

[13] N. Wiener, *Nonlinear Problems in Random Theory*, M.I.T. Press, Cambridge, MA, 1958.

[14] S. Boyd, *Volterra Series: Engineering Fundamentals*, Ph.D. dissertation, University of California, Berkeley, 1985.

[15] W. Rugh, *Nonlinear System Theory, The Volterra-Wiener Approach*, The Johns Hopkins University Press, Baltimore, MD, 1981.

[16] M. Schetzen, *The Volterra and Wiener Theories of Nonlinear Systems*, John Wiley & Sons, New York, 1980.

[17] A. French, S. Sekizawa, U. Hoger, and P. Torkkeli, "Predicting the responses of mechanoreceptor neurons to physiological inputs by nonlinear system identification," *Annals of Biomedical Engineering*, vol. 29, pp. 187–194, 2001.

[18] A.S. French and M.J. Korenberg, "Dissection of a nonlinear cascade model for sensory encoding," *Annals of Biomedical Engineering*, vol. 19, pp. 473–484, 1991.

[19] V. Marmarelis, M. Juusola, and A. French, "Principal dynamic mode analysis of nonlinear transduction in a spider mechanoreceptor," *Annals of Biomedical Engineering*, vol. 27, pp. 391–402, 1999.

[20] V.Z. Marmarelis, "Wiener analysis of nonlinear feedback in sensory systems," *Annals of Biomedical Engineering*, vol. 19, pp. 345–382, 1991.

[21] M. Iatrou, T.W. Berger, and V.Z. Marmarelis, "Application of a novel modeling method to the nonstationary properties of potentiation in the rabbit hippocampus," *Annals of Biomedical Engineering*, vol. 27, pp. 581–591, 1999.

[22] Q. Zhang, B. Suki, D. Westwick, and K. Lutchen, "Factors affecting Volterra kernel estimation: Emphasis on lung tissue viscoelasticity," *Annals of Biomedical Engineering*, vol. 26, pp. 103–116, 1998.

[23] K. Chon, N. Holstein-Rathlou, D. Marsh, and V. Marmarelis, "Comparative nonlinear modeling of renal autoregulation in rats: Volterra approach versus artificial neural networks," *IEEE Trans. on Neural Networks*, vol. 9, no. 3, pp. 430–435, May 1998.

[24] K. Chon, Y. Chen, N. Holstein-Rathlou, and V. Marmarelis, "Nonlinear system analysis of renal autoregulation in normotensive and hypertensive rats," *IEEE Trans. on Biomedical Engineering*, vol. 45, no. 3, pp. 342–353, March 1998.

[25] V. Marmarelis, K. Chon, Y. Chen, D. Marsh, and N. Holstein-Rathlou, "Nonlinear analysis of renal autoregulation under broadband forcing conditions," *Annals of Biomedical Engineering*, vol. 21, pp. 591–603, 1993.

[26] A.H. Hoffman and P. Grigg, "Using uniaxial pseudorandom stress stimuli to develop soft tissue constitutive equations," *Annals of Biomedical Engineering*, vol. 30, pp. 44–53, 2002.

[27] W. Silva, "Application of nonlinear systems theory to transonic unsteady aerodynamic responses," *Journal of Aircraft*, vol. 30, no. 5, pp. 660–668, Sept.-Oct. 1993.

[28] W. Silva, *Discrete-Time Linear and Nonlinear Aerodynamic Impulse Responses for Efficient CFD Analyses*, Ph.D. dissertation, The College of William and Mary, Williamsburg, VA, 1997.

[29] D.E. Raveh, "Reduced-order models for nonlinear unsteady aerodynamics," *AIAA Journal*, vol. 39, no. 8, pp. 1417–1429, August 2001.

[30] D.E. Raveh, Y. Levy, and M. Karpel, "Efficient aeroelastic analysis using computational unsteady aerodynamics," *Journal of Aircraft*, vol. 38, no. 3, pp. 547–556, May-June 2001.

[31] P. Marzocca, L. Librescu, and W. Silva, "Volterra series approach for nonlinear aeroelastic response of 2-d lifting surfaces," *AIAA-2001-2459*, 2001.

[32] P. Marzocca, L. Librescu, and W.A. Silva, "Aeroelastic response of nonlinear wing sections using a functional series technique," *AIAA Journal*, vol. 40, no. 5, pp. 813–823, May 2002.

[33] R. Prazenica, R. Lind, and A. Kurdila, "Uncertainty estimation from Volterra kernels for robust flutter analysis," *Journal of Guidance, Control, and Dynamics*, to appear.

[34] P. Reisenthel, "Prediction of unsteady aerodynamic forces via nonlinear kernel identification," *CEAS/AIAA/ICASE/NASA Langley International Forum on Aeroelasticity and Structural Dynamics*, June 1999.

[35] G.A. Tsihrintzis and A.J. Devaney, "A Volterra series approach to nonlinear traveltime tomography," *IEEE Trans. on Geoscience and Remote Sensing*, vol. 38, no. 4, pp. 1733–1742, July 2000.

[36] G.A. Tsihrintzis and A.J. Devaney, "Higher order (nonlinear) diffraction tomography: Inversion of the Rytov series," *IEEE Trans. on Information Theory*, vol. 46, no. 5, pp. 1748–1761, August 2000.

[37] W.S. Gray and B. Nabet, "Volterra series analysis and synthesis of a neural network for velocity estimation," *IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics*, vol. 29, no. 2, pp. 190–197, April 1999.

[38] K. Worden and G.L. Tomlinson, *Nonlinearity in Structural Dynamics: Detection, Identification, and Modelling*, Institute of Physics Publishing, Philadelphia, 2001.

[39] I.A. Sibetheros, O.R. Rijken, and J.M. Niedzwecki, "Volterra series-based system analysis of random wave interaction with a horizontal cylinder," *Ocean Engineering*, vol. 27, pp. 241–270, 2000.

[40] D. Aiordachioaie, E. Ceanga, R. De Keyser, and Y. Naka, "Detection and classification of non-linearities based on Volterra kernels processing," *Engineering Applications of Artificial Intelligence*, vol. 14, pp. 497–503, 2001.

[41] E. Kuruoglu, P. Rayner, and W. Fitzgerald, "Least l_p-norm impulsive noise cancellation with polynomial filters," *Signal Processing*, vol. 69, pp. 1–14, 1998.

[42] K.V. Peddanarappagari and M. Brandt-Pearce, "Volterra series approach for optimizing fiber-optic communications system designs," *Journal of Lightwave Technology*, vol. 16, no. 11, pp. 2046–2055, November 1998.

[43] K.M. Cheng and S. Chan, "Reduction of intermodulation distortion in microwave active bandpass filters - theory and experiments," *IEEE Trans. on Microwave Theory and Techniques*, vol. 48, no. 2, pp. 221–225, February 2000.

[44] K.L. Fong, "High-frequency analysis of linearity improvement technique of common-emitter transconductance stage using a low-frequency-trap network," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 8, pp. 1249–1252, August 2000.

[45] A. Ahmed, S. Islam, and A. Anwar, "A temperature-dependent nonlinear analysis of GaN/AlGaN HEMTs using Volterra series," *IEEE Trans. on Microwave Theory and Techniques*, vol. 49, no. 9, pp. 1518–1524, September 2001.

[46] J.A. Cherry and W.M. Snelgrove, "On the characterization and reduction of distortion in bandpass filters," *IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications*, vol. 45, no. 5, pp. 523–537, May 1998.

[47] C. Cheng and E.J. Powers, "Optimal Volterra kernel estimation algorithms for a nonlinear communication system for PSK and QAM inputs," *IEEE Trans. on Signal Processing*, vol. 49, no. 1, pp. 147–163, January 2001.

[48] A. Borys, *Nonlinear Aspects of Telecommunications: Discrete Volterra Series and Nonlinear Echo Cancellation*, CRC Press, New York, 2001.

[49] V. Parsa, P. Parker, and R. Scott, "Adaptive stimulus artifact reduction in noncortical somatosensory evoked potential studies," *IEEE Trans. on Biomedical Engineering*, vol. 45, no. 2, pp. 165–179, February 1998.

[50] P. Strauch and B. Mulgrew, "Active control of nonlinear noise processes in a linear duct," *IEEE Trans. on Signal Processing*, vol. 46, no. 9, pp. 2404–2412, September 1998.

[51] L. Tan and J. Jiang, "Adaptive Volterra filters for active control of nonlinear noise processes," *IEEE Trans. on Signal Processing*, vol. 49, no. 8, pp. 1667–1676, August 2001.

[52] F.J. Doyle III, R.K. Pearson, and B.A. Ogunnaike, *Identification and Control Using Volterra Models*, Springer, London, 2002.

[53] F.J. Doyle III, B.A. Ogunnaike, and R.K. Pearson, "Nonlinear model-based control using second-order Volterra models," *Automatica*, vol. 31, no. 5, pp. 697–714, 1995.

[54] B. Maner, F. Doyle III, B. Ogunnaike, and R. Pearson, "Nonlinear model predictive control of a simulated multivariable polymerization reactor using second-order Volterra models," *Automatica*, vol. 32, no. 9, pp. 1285–1301, 1996.

[55] R.K. Pearson, B.A. Ogunnaike, and F.J. Doyle III, "Identification of structurally constrained second-order Volterra models," *IEEE Trans. on Signal Processing*, vol. 44, no. 11, pp. 2837–2846, November 1996.

[56] R. Madhavan, "Tracking control of a pendulum using a numerically efficient state estimation approach for polynomic non-linear systems," *Int. Journal of Control*, vol. 72, no. 17, pp. 1565–1591, 1999.

[57] O. Yamanaka, H. Ohmori, and A. Sano, "Design method of exact model matching control for finite Volterra series systems," *Int. Journal of Control*, vol. 68, no. 1, pp. 107–124, 1997.

[58] Y.W. Lee and M. Schetzen, "Measuring of the Wiener kernels of a non-linear system by cross-correlation," *International Journal of Control*, vol. 2, no. 3, pp. 237–254, 1965.

[59] S. Boyd, Y.S. Tang, and L.A. Chua, "Measuring Volterra kernels," *IEEE Trans. on Circuits and Systems*, vol. 30, no. 8, pp. 571–577, 1983.

[60] V.Z. Marmarelis, "Identification of nonlinear biological systems using Laguerre expansions of kernels," *Annals of Biomedical Engineering*, vol. 21, pp. 573–589, 1993.

[61] M. Nikolaou and D. Mantha, "Efficient nonlinear modeling using wavelets and related compression techniques," *NSF Workshop on Nonlinear Model Predictive Control*, 1998.

[62] A. Kurdila, R. Prazenica, O. Rediniotis, and T. Strganac, "Multiresolution methods for reduced-order models for dynamical systems," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 2, pp. 193–200, Mar.-Apr. 2001.

[63] J. Wray and G. Green, "Calculation of the Volterra kernels of non-linear dynamic systems using an artificial neural network," *Biological Cybernetics*, vol. 71, no. 3, pp. 187–195, 1994.

[64] E.P. Wigner, "On the quantum correction for thermodynamic equilibrium," *Physical Review*, vol. 40, pp. 749–759, 1932.

[65] D. Gabor, "Theory of communications," *J. IEE*, vol. 93, pp. 429–457, 1946.

[66] A. Croiser, D. Esteban, and C. Galand, "Perfect channel splitting by use of interpolation/decimation/tree decomposition techniques," in *Int. Conference on Information Sciences and Systems*, Patras, Greece, August 1976, pp. 443–446.

[67] R.E. Crochiere, S.A. Webber, and J.L. Flanagan, "Digital coding of speech in subbands," *Bell System Technical Journal*, vol. 55, no. 8, pp. 1069–1085, October 1976.

[68] M.J. Smith and T.P. Barnwell III, "A procedure for designing exact reconstruction filter banks for tree structured subband coders," in *Proc. IEEE Int. Conference on Acoustics, Speech, and Signal Processing*, San Diego, CA, March 1984.

[69] Y. Meyer, *Wavelets and Operators*, Cambridge University Press, New York, 1992.

[70] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Comm. Pure and Applied Mathematics*, vol. 41, pp. 906–996, 1988.

[71] A. Cohen, I. Daubechies, and J.C. Feauveau, "Biorthogonal bases of compactly supported wavelets," *Comm. Pure and Applied Mathematics*, vol. 45, pp. 485–559, 1992.

[72] G. Donovan, J. Geronimo, D. Hardin, and P. Massopust, "Construction of orthogonal wavelets using fractal interpolation functions," *SIAM Journal of Mathematical Analysis*, vol. 27, no. 4, pp. 1158–1192, July 1996.

[73] J. Geronimo, D. Hardin, and P. Massopust, "Fractal functions and wavelet expansions based on several scaling functions," *Journal of Approximation Theory*, vol. 78, pp. 373–401, 1994.

[74] I. Daubechies, *Ten Lectures on Wavelets*, Capital City Press, Montpelier, VT, 1992.

[75] R. Ogden, *Essential Wavelets for Statistical Applications and Data Analysis*, Birkhauser, Boston, MA, 1997.

[76] C.K. Chui, *An Introduction to Wavelets*, Academic Press, Boston, MA, 1992.

[77] C.K. Chui, *Wavelets: A Tutorial in Theory and Applications*, Academic Press, Boston, MA, 1992.

[78] P. Wojtaszczyk, *A Mathematical Introduction to Wavelets*, Cambridge University Press, Cambridge, MA, 1997.

[79] P. Goupillaud, A. Grossman, and J. Morlet, "Cycle-octave and related transforms in seismic signal analysis," *Geoexploration*, vol. 23, no. 1, pp. 85–102, October 1984.

[80] S.G. Mallat, "Multiresolution approximation and wavelet orthonormal bases of $L^2(R)$," *Trans. of the American Mathematical Society*, vol. 315, no. 1, pp. 69–87, September 1989.

[81] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, Boston, MA, 1998.

[82] G. Strang and T. Nguyen, *Wavelets and Filter Banks*, Wellesley-Cambridge Press, Wellesley, MA, 1996.

[83] M. Vetterli and J. Kovacevic, *Wavelets and Subband Coding*, Prentice Hall, Englewood Cliffs, NJ, 1995.

[84] C. Burrus, R. Gopinath, and H. Guo, *Introduction to Wavelets and Wavelet Transforms: A Primer*, Prentice Hall, Upper Saddle River, NJ, 1998.

[85] W. Dahmen, S. Prossdorf, and R. Schneider, "Multiscale methods for pseudo-differential equations on smooth closed manifolds," in *Wavelets: Theory, Algorithms, and Applications*, C. Chui, L. Montefusco, and L. Puccio, Eds., pp. 385–424. Academic Press, Boston, MA, 1994.

[86] S. Bertoluzza, G. Naldi, and J. Ravel, "Wavelet methods for the numerical solution of boundary value problems on the interval," in *Wavelets: Theory, Algorithms, and Applications*, C. Chui, L. Montefusco, and L. Puccio, Eds., pp. 425–448. Academic Press, Boston, MA, 1994.

[87] W. Dahmen, A. Kurdila, and P. Oswald, *Multiscale Wavelet Methods for Partial Differential Equations*, Academic Press, Boston, MA, 1997.

[88] D.L. Donoho, "Unconditional bases are optimal bases for data compression and for statistical estimation," *Applied and Computational Harmonic Analysis*, vol. 1, no. 1, pp. 100–115, 1993.

[89] M. Farge, "Wavelet transforms and their application to turbulence," *Annual Review of Fluid Mechanics*, vol. 24, pp. 395–457, 1994.

[90] J. Ko, A. Kurdila, J. Gilarranz, and O. Rediniotis, "Particle image velocimetry via wavelet analysis," *AIAA Journal*, vol. 36, no. 8, pp. 1451–1459, 1998.

[91] J. Ko, A. Kurdila, and O. Rediniotis, "Divergence-free bases and multiresolution methods for reduced-order flow modeling," *AIAA Journal*, vol. 38, no. 12, pp. 2219–2232, December 2000.

[92] M. Akay and A. Daubenspeck, "Respiratory related evoked responses to graduated pressure pulses using wavelet transform methods," *Annals of Biomedical Engineering*, vol. 28, pp. 1126–1135, 2000.

[93] R. Rubini and U. Meneghetti, "Application of the envelope and wavelet transform analysis for the diagnosis of incipient faults in ball bearings," *Mechanical Systems and Signal Processing*, vol. 15, no. 2, pp. 287–302, 2001.

[94] R. Lind, K. Snyder, and M. Brenner, "Wavelet analysis to characterise nonlinearities and predict limit cycles of an aeroelastic system," *Mechanical Systems and Signal Processing*, vol. 15, no. 2, pp. 337–356, 2001.

[95] W.J. Staszewski and J.E. Cooper, "Wavelet approach to flutter data analysis," *Journal of Aircraft*, vol. 39, no. 1, pp. 125–132, January-February 2002.

[96] D.T. Westwick and R.E. Kearney, "Nonparametric identification of nonlinear biomedical systems, part I: theory," *Critical Reviews in Biomedical Engineering*, vol. 26, no. 3, pp. 153–226, 1998.

[97] I. Sandberg, "Bounds for discrete-time Volterra series representations," *IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications*, vol. 46, no. 1, pp. 135–139, January 1999.

[98] I. Sandberg, "On Volterra expansions for time-varying nonlinear systems," *IEEE Trans. on Circuits and Systems*, vol. CAS-30, no. 2, pp. 61–67, February 1983.

[99] D. Walnut, *An Introduction to Wavelet Analysis*, Birkhauser, Boston, 2002.

[100] T. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1987.

[101] Z. Chen, C. Micchelli, and Y. Xu, "A construction of interpolating wavelets on invariant sets," *Mathematics of Computation*, vol. 68, no. 228, pp. 1569–1587, March 1999.

[102] I. Tawfig and T. Vinh, "Study of non-linear structure using an electronic slave hammer," *Mechanical Systems and Signal Processing*, vol. 14, no. 6, pp. 847–870, 2000.

[103] E. Bedrosian and S.O. Rice, "The output properties of Volterra systems driven by harmonic and Gaussian inputs," *Proceedings of the IEEE*, vol. 59, pp. 1688–1707, 1971.

[104] K. Worden, G. Manson, and G.R. Tomlinson, "A harmonic probing algorithm for the multi-input Volterra series," *Journal of Sound and Vibration*, vol. 201, no. 1, pp. 67–84, 1997.

[105] C. Evans, D. Rees, L. Jones, and M. Weiss, "Periodic signals for measuring nonlinear Volterra kernels," *IEEE Transactions on Instrumentation and Measurement*, vol. 45, no. 2, pp. 362–371, April 1996.

[106] M. Weiss, C. Evans, and D. Rees, "Identification of nonlinear cascade systems using paired multisine signals," *IEEE Transactions on Instrumentation and Measurement*, vol. 47, no. 1, pp. 332–336, February 1998.

[107] M.J. Korenberg and I.W. Hunter, "The identification of nonlinear biological systems: Volterra kernel approaches," *Annals of Biomedical Engineering*, vol. 24, no. 4, pp. 250–268, 1996.

[108] M.J. Korenberg, "Parallel cascade identification and kernel estimation for nonlinear systems," *Annals of Biomedical Engineering*, vol. 19, pp. 429–455, 1991.

[109] M.J. Korenberg, "Identifying nonlinear difference equations and functional expansion representations: the fast orthogonal algorithm," *Annals of Biomedical Engineering*, vol. 16, pp. 123–142, 1988.

[110] G.P. Liu, V. Kadirkamanathan, and S.A. Billings, "On-line identification of nonlinear systems using Volterra polynomial basis function neural networks," *Neural Networks*, vol. 11, pp. 1645–1657, 1998.

[111] D.C. Lay, *Linear Algebra and Its Applications*, Addison-Wesley, New York, second edition, 1997.

[112] D. Colton and R. Kress, *Inverse Acoustic and Electromagnetic Scattering Theory*, Springer, New York, second edition, 1998.

[113] C.W. Groetsch, *Inverse Problems in the Mathematical Sciences*, Vieweg, Wiesbaden, 1993.

[114] P.C. Hansen, *Rank Deficient and Discrete Ill-Posed Problems*, SIAM, Philadelphia, 1998.

[115] G. Strang, *Linear Algebra and Its Applications*, Harcourt Brace Jovanovich, San Diego, third edition, 1988.

[116] R. Craig, *Structural Dynamics: An Introduction To Computer Methods*, John Wiley & Sons, New York, 1981.

[117] M.W. Kehoe, "A historical overview of flight flutter testing," *NASA-TM-4720*, October 1995.

[118] N.H. Zimmerman and J.T. Weissenburger, "Prediction of flutter onset speed based on flight testing at subcritical speeds," *Journal of Aircraft*, vol. 1, no. 4, pp. 190–202, July-August 1964.

[119] R. Lind, "Flight testing with the flutterometer," *Journal of Aircraft*, to appear.

[120] D. Voracek, M. Reaves, L. Horta, and S. Potter, "Actuators for ground and flight test structural excitation," *AIAA-2002-1349*, 2002.

BIOGRAPHICAL SKETCH

Richard (Chad) Prazenica was born in Cranston, RI, in 1973. In May of 1995, he received the Bachelor of Science in Engineering (BSE) degree from the Department of Mechanical Engineering and Applied Mechanics at the University of Pennsylvania. In August of 1999, he received the Master of Science degree from the Department of Aerospace Engineering, Mechanics and Engineering Science at the University of Florida. In August of 2002, he will receive the Doctor of Philosophy degree in aerospace engineering from the University of Florida. After graduation, Chad plans to work at the NASA Dryden Flight Research Center where he has been awarded a National Research Council (NRC) Postdoctoral Fellowship.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Dr. Andrew J. Kurdila, Chair
Professor of Aerospace Engineering, Mechanics and Engineering Science

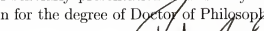
I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Dr. Louis N. Cattafesta
Assistant Professor of Aerospace Engineering, Mechanics and Engineering Science

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

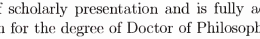
Dr. Ulrich H. Kurzweg
Professor of Aerospace Engineering, Mechanics and Engineering Science

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Dr. Richard C. Lind
Assistant Professor of Aerospace Engineering, Mechanics and Engineering Science

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.



Dr. Shari Moskow
Assistant Professor of Mathematics

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August 2002

Pramod P. Khargonekar
Dean, College of Engineering

Winfred M. Phillips
Dean, Graduate School